2007-03

Velocity estimation using forward looking sonar

Dolbec, Michael R.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

VELOCITY ESTIMATION USING FORWARD LOOKING SONAR

by

Michael R. Dolbec

March 2007

Thesis Advisor:	Doug Horner
Associate Advisor:	Mathias Kölsch

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2007	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Velocity Estimation Using Forward Looking Sonar.			**5. FUNDING NUMBERS**
6. AUTHOR: Dolbec, Michael			
7. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME AND ADDRESS N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE**

13. ABSTRACT

The thesis investigates a method to estimate the forward velocity and heading rate of an autonomous underwater vehicle (AUV). Through relatively new technologies small AUVs are now able to mount a Forward Looking Sonar (FLS) on the vehicle's nose. This can be used for obstacle avoidance and feature based navigation. The sensor can also be used to estimate motion of the AUV, which can be useful for undersea navigation. The thesis focuses on a template matching technique used in computer vision. Two sequential sonar images are compared with the goal of finding the rotation and translation that best correlates the first to the second sonar image. The transformation which maximizes the correlation coefficient is then converted to forward velocity and heading rate through motion analysis.

Experimentation shows that the method provides accurate estimates for both the forward velocity and heading rate of the AUV. Accuracy of the estimates for forward velocity was at the limitation of the resolution of the sonar. Using velocities estimated through image processing applied to FLS images entirely with software, the weight and energy resources currently required by standard measurement techniques could be used to increase the vehicles endurance or for additional payload capacity. Another benefit would be the reduction in acoustic and electrical interference with the FLS and side scan sonar, which would improve the vehicle's obstacle avoidance and mine-hunting capability. The vehicle could become more flexible in its capability to support additional roles vice specific missions. This method holds the promise for permitting smaller AUVs with a FLS to navigate undersea more accurately.

14. SUBJECT TERMS Computer vision, Unmanned Underwater Vehicle, Autonomous Underwater Vehicles, REMUS, Velocity Estimation			**15. NUMBER OF PAGES** 134
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

VELOCITY ESTIMATION USING FORWARD LOOKING SONAR

Michael R. Dolbec
Lieutenant, United States Navy
B.S., Maine Maritime Academy, 2000

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2007

Author: Michael R. Dolbec

Approved by: Doug Horner
Thesis Advisor

Mathias Kölsch
Associate Advisor

Anthony J. Healey
Chair, Department of Mechanical and
Astronautical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The thesis investigates a method to estimate the forward velocity and heading rate of an autonomous underwater vehicle (AUV). Through relatively new technologies small AUVs are now able to mount a Forward Looking Sonar (FLS) on the vehicle's nose. This can be used for obstacle avoidance and feature based navigation. The sensor can also be used to estimate motion of the AUV, which can be useful for undersea navigation. The thesis focuses on a template matching technique used in computer vision. Two sequential sonar images are compared with the goal of finding the rotation and translation that best correlates the first to the second sonar image. The transformation which maximizes the correlation coefficient is then converted to forward velocity and heading rate through motion analysis.

Experimentation shows that the method provides accurate estimates for both the forward velocity and heading rate of the AUV. Accuracy of the estimates for forward velocity was at the limitation of the resolution of the sonar. Using velocities estimated through image processing applied to FLS images entirely with software, the weight and energy resources currently required by standard measurement techniques could be used to increase the vehicles endurance or for additional payload capacity. Another benefit would be the reduction in acoustic and electrical interference with the FLS and side scan sonar, which would improve the vehicle's obstacle avoidance and mine-hunting capability. The vehicle could become more flexible in its capability to support additional roles vice specific missions. This method holds the promise for permitting smaller AUVs with a FLS to navigate undersea more accurately.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION 1

A. GENERAL 1
B. MOTIVATION AND RELEVANCE 2
C. REMUS VEHICLE DESCRIPTION 4
D. THESIS SCOPE AND STRUCTURE 7

II. ACTIVE SONAR 9

A. INTRODUCTION 9
B. ACTIVE SONAR EQUATIONS 10
C. DOPPLER VELOCITY MEASUREMENTS 12
D. NOISE ASSOCIATED WITH SONAR SYSTEMS 14
E. BLAZED ARRAY TRANSDUCERS 16
F. NEAR-FIELD EFFECTS 20

III. COMPUTER VISION 23

A. INTRODUCTION 23
B. MOTION ANALYSIS 24
C. FEATURE IDENTIFICATION 26
D. EUCLIDEAN TRANSFORMATION 27
E. CORRELATION COEFFICIENT BASED MATCHING 30

IV. STEERING MODEL 35

A. INTRODUCTION 35
B. EQUATIONS OF MOTION IN THE HORIZONTAL PLANE 35
C. HYDRODYNAMIC COEFFICIENTS 39
D. VEHICLE KINEMATICS 41
E. VEHICLE DYNAMICS 42
F. APPLICATION 42

V. SIMULATION AND RESULTS 45

A. INTRODUCTION 45
B. MODEL PROCESS 46
1. Image Preprocessing 48
2. Transformation 51
3. Correlation coefficient 56
4. Motion Analysis 56
C. RESULTS 56
1. REMUS AUV Mission 012506 57
2. Lateral Velocity and Heading Rate Experiments 022307 65

VI. CONCLUSIONS AND RECOMMENDATIONS 67

A. CONCLUSIONS 67

B. RECOMMENDATIONS FOR FUTURE WORK 67

APPENDIX A: SIMULATION RESULTS 73

APPENDIX B: MATLAB CODE FOR VELOCITY ESTIMATES 75

APPENDIX C: MATLAB CODE FOR OPENING FLS IMAGES 107

APPENDIX D: MATLAB CODE FOR SEGMENTATION LINES 109

LIST OF REFERENCES 111

BIBLIOGRAPHY 113

INITIAL DISTRIBUTION LIST 115

LIST OF FIGURES

Figure 1. Tactical Application of REMUS AUVs Deployed in Operation Iraqi Freedom (From UUV Master Plan, 2004) 2
Figure 2. REMUS AUV (From Hydroid Inc., 2007) 5
Figure 3. REMUS AUV (From UUV Master Plan, 2004) 8
Figure 4. Blazed Array FLS Image of Multiple Features REMUS AUV 012506. 10
Figure 5. REMUS ADCP Upward Beams (From NOAA's Marine Navigation 2007) 12
Figure 6. Naval Postgraduate School REMUS AUV with BLUEVIEW FLS 16
Figure 7. Naval Postgraduate School REMUS AUV with BLUEVIEW FLS. Note the nose cone is removed, and the transducers are in the horizontal configuration. 17
Figure 8. Field of View of a Blazed Array transducer (From BlueView Technologies Inc., 2007) 17
Figure 9. Illustration of Blazed Array Beamforming (From Thompson, 2001) 19
Figure 10. FLS Image with Dominate Near-Field Effects REMUS AUV 012506. 21
Figure 11. Rigid Transformation Model of Image Displacement 29
Figure 12. Example of a pattern in FLS image from REMUS AUV 012506 31
Figure 13. Local and Global Coordinate System (From Marco and Healey, 2001) 36
Figure 14. Using Sensor Updates to Update State Information (From Smith, Self, and Cheeseman, 1990) 43
Figure 15. REMUS AUV Position and Feature Positions 012506 47
Figure 16. Correlation coefficient versus Forward Velocity and Heading Rate from two sequential FLS images REMUS AUV 012506 49
Figure 17. Correlation coefficient versus Forward Velocity and Heading Rate from two sequential FLS images REMUS AUV 012506 50
Figure 18. Example of Translation Being Applied to a single FLS Image. Upper left image shows the initial sonar image with the required FOV outlined. Lower right shows the sonar images with the translation applied. Note that the near-field effects have not been removed so the process can be more easily observed. 52
Figure 19. Removing Overlapping Pixels in Preparation for Counter-Clockwise Rotation Being Applied to two FLS Images. The upper left sonar image is the previous image with translation applied, while the upper right is the current sonar image. The white outlines the two FOV to be compared. The bottom images show the FOV with overlapping pixels removed. Note that the near-field effects have not been removed so the process can be more easily observed. 53
Figure 20. Removing Overlapping Pixels in Preparation for Clockwise Rotation Being Applied to two FLS Images. The upper left sonar image is the previous image with translation applied, while the upper right is the current sonar image. The white outlines the two FOV to be compared. The bottom images show the FOV with overlapping pixels removed. Note that

the near-field effects have not been removed so the process can be more easily observed. 54
Figure 21. Image Rotation and Removing Overlap to Ensure Proper Calculation of the Correlation Coefficient. The upper two sonar images are to be compared. The second row shows the right sonar image rotated. The third row shows the addition of rows and columns of zeros to match the two images origins. The final row shows the addition of rows and columns of zeros to create two identically sized matrices. Note that the near-field effects have not been removed so the process can be more easily observed. ..55
Figure 22. Estimated and Measured U for REMUS 012506 from 31627 to 31849 57
Figure 23. Estimated and Measured U for REMUS 012506 from 31850 to 32063. Note that the first circle identifies when the AUV was in the minefield, the second is identifies when the AUV conducted two ninety degree turns 58
Figure 24. Estimated and Measured U for REMUS 012506 from 32064 to 32279. Note that the first circle identifies when the AUV was in the minefield, the second is identifies when the AUV conducted two ninety degree turns 58
Figure 25. Estimated and Measured U for REMUS 012506 from 32280 to 32491 59
Figure 26. Estimated and Measured U for REMUS 012506 from 32492 to 32710 59
Figure 27. Estimated and Measured U for REMUS 012506 from 32711 to 32919 60
Figure 28. Estimated and Measured U for REMUS 012506 from 32920 to 33129 60
Figure 29. Estimated and Measured U for REMUS 012506 from 33130 to 33344 61
Figure 30. Estimated and Measured U for REMUS 012506 from 33345 to 33559 61
Figure 31. Estimated and Measured U for REMUS 012506 from 33561 to 33779 62
Figure 32. Estimated and Measured U for REMUS 012506 from 33780 to 33996 62
Figure 33. Estimated and Measured U for REMUS 012506 from 33997 to 34229 63
Figure 34. Unbounded Estimated and Measured U for Initial Deployment REMUS 012506 64
Figure 35. Simulated Heading Rate for REMUS 022307 66
Figure 36. Blazed Array FLS Image of Multiple Features REMUS AUV 012506 69
Figure 37. Geometric Relationship between Multiple Features 70

LIST OF TABLES

Table 1. REMUS Specifications (From Hydroid Inc., 2007) 6
Table 2. Blue View FLS Specifications (From Blue View Inc., 2007) 18
Table 3. REMUS Hydrodynamic Coefficients for Steering (From Fodrea, 2002).....41
Table 4. Example Structure Containing Data from REMUS Mission 012506.....48

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ADCP	Acoustic Doppler Current Profiler
ARIES	Acoustic Radio Interactive Exploratory Server
AUV	Autonomous Underwater Vehicle
DVL	Doppler Velocity Log
EKF	Extended Kalman Filter
EOM	Equations of Motion
FOV	Field of View
FLS	Forward Looking Sonar
GPS	Global Positioning System
GWOT	Global War on Terrorism
ISR	Intelligence, Surveillance, and Reconnaissance
LBL	Long Base-Line
NL	Noise Level
REMUS	Remote Environmental Monitoring Units
RDI	RD Instruments
SL	Source Level
SLAM	Simultaneous Localization and Mapping
SNR	Signal to Noise Ratio
TL	Transmission Loss
TS	Target Strength
UUV	Unmanned Underwater Vehicle

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I thank God first and foremost for the blessings and opportunities He has presented in my life. I would also like to thank my family. My wonderful wife, Katherine, whose support and encouragement was always uplifting; I thank you for your patience and understanding through the long hours. To my parents, I thank you for instilling within me the desire and determination to continue learning.

I would like to thank my thesis advisors, Professor Doug Horner, for his expert insight, direction, and assistance during the development of this research. Professor Mathias Kölsch provided outstanding instruction in image processing techniques. Both advisors are dedicated professionals whose knowledge and experience were invaluable during this research.

I would also like to make special mention of Professor Anthony Healey and all of the personnel who contribute to the Naval Postgraduate School Center for Autonomous Unmanned Vehicle Research. I thank you for listening, providing input, and offering thoughts and opinions during the course of this work.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. GENERAL

The application of unmanned vehicles in both civilian and military roles continues to expand and grow as new capabilities are demonstrated. The use of unmanned vehicles as force multipliers and also as risk reducers has been directed within Sea Power 21. In the current Global War on Terrorism (GWOT) the use of unmanned vehicles in military roles has been rapidly evolving. The vision for these unmanned vehicles includes roles such as Intelligence, Surveillance, and Reconnaissance (ISR) as well as improvised explosive devises/mine countermeasures. Specific missions, such as minefield detection and clearance as well as improvised explosive devise disposal are roles that are perfectly suited for the unmanned vehicles, as it reduces risk to personnel. In some instances, such as mine hunting, the unmanned vehicles are capable of performing the role faster and with greater accuracy than humans. Due to the vastly different environments in which they operate, unmanned vehicles are designed for specific missions. Due to the mission and situation, the amount of input necessary from a human operator will vary greatly.

The Navy Unmanned Underwater Vehicle (UUV) Master Plan (Department of the Navy, 2004) identifies several of the areas where research and development continues to be required. The development of autonomy and control as well as sensors and sensor processing are areas requiring major research. Energy and propulsion as well as navigation and communication also continue to be areas where research and growth are required and are also specifically identified within the UUV Master Plan. Sea Power 21 has specifically identified unmanned systems within the future vision of the U.S. Navy. To ensure that the U.S. Navy maintains sea superiority, the development and employment of the technologies surrounding the unmanned vehicles must continue at a pace to meet the expected roles. The immediate needs of the military, involve unmanned vehicles conducting mine countermeasure operations. For example several Remote Environmental Monitoring Units (REMUS) Autonomous Underwater Vehicles (AUVs) were employed during Operation Iraqi Freedom (Figure 1) to assist in the clearance of

mines within the harbor of Umm Qasr. With the REMUS AUVs operating in cooperation with additional mine clearance assets safe lanes of passage were quickly established for the arrival of humanitarian aide.

Figure 1. Tactical Application of REMUS AUVs Deployed in Operation Iraqi Freedom (From UUV Master Plan, 2004)

B. MOTIVATION AND RELEVANCE

As the AUVs roles in the battlespace become more prevalent researchers must examine the current limitations of the vehicles. Within the U.S. Navy's UUV Master Plan the continued research and development of sensor processing and navigation are specifically identified. Increased intelligent autonomy is necessary to allow unmanned systems to operate independent from human input for extended periods on more complex tasks. Autonomous vehicles must be able to collect, evaluate, and sort data based on mission performance and priorities. UUVs require significantly more sophisticated autonomy since maintaining a communications link between the vehicle and a human operator is often impossible.

For aerial, ground, and surface unmanned vehicles the challenge of navigation can be resolved by incorporating the inputs from the Global Positioning Satellites (GPS). Many AUVs also employ these inputs; however the GPS inputs are only available when the vehicle or GPS antenna is above the surface of the water. When the vehicle is operating in the undersea environment it relies upon additional inputs from equipment such as long base-line (LBL) transponders, accelerometers, and gyroscopes to track how the vehicle has moved from the last known position. Another method of measuring velocities in the forward and lateral directions involves the use of active sonar. The Acoustic Doppler Current Profiler (ADCP) uses this method, and the hardware adds weight to the vehicle and requires energy to operate. If the velocities could be estimated through image processing applied to legacy sonar images entirely with software, the weight and energy resources currently required by the ADCP could be used to increase the vehicles endurance or for additional payload capacity. Another benefit would be that there would be a reduction in acoustic and electrical interference with the FLS and side scan sonar, which would improve the vehicle's obstacle avoidance and mine-hunting capability. The vehicle could become more flexible in its capability to support additional roles vice specific missions. For example the additional payload capacity could be used to carry mine countermeasure neutralization charges or deployable sensor/communication arrays to support antisubmarine warfare or ISR operations. The U.S. Navy's UUV Master Plan also identifies the need for small, man-portable AUVs. Small AUVs may not be capable of supporting the larger hardware such as the ADCP therefore an alternative is required. For one time use AUVs, such a vehicle used to deploy mine neutralization charges, capital costs would need to be minimized. Therefore an alternative to the ADCP would be desired.

Active sonar is one method that is used to measure the relative position of features in an undersea environment. Feature detectors are used to extract the features from the sonar images and the relative positions, range and bearing can be determined. The relative position of the features is used in a position estimation filter, such as an Extended Kalman Filter (EKF) to determine the updated position of the unmanned vehicle in the undersea environment. This is called feature-based navigation. In some situations there

are multiple features within the sonar field of view (FOV). The proposed method can prevent confusing the features and sending bad inputs into the EKF, which would then result in erroneous estimates of the vehicles position in the environment. Using velocity estimates from the sonar images accurate predictions of the location of features from one image to the next can be determined. Using this predicted position, compared to the measured position, ensures that individual features are tracked accordingly. Ensuring that the new positions of the features are inputted correctly will result in a more accurate vehicle position update.

C. REMUS VEHICLE DESCRIPTION

The Remote Environmental Monitoring Units (REMUS) AUV was used to evaluate this thesis. The advantage of the Naval Postgraduate School REMUS AUV is that it is equipped with both the forward looking sonar (FLS) and an acoustic Doppler current profiler (ADCP) Doppler velocity log (DVL). The proposed image correlation method from the FLS can be evaluated against the results of the ADCP DVL measurements. This permits comparisons between the estimated and measured velocities to ensure the algorithm runs correctly.

REMUS are commercially built low cost AUVs. They are small, lightweight AUVs which were originally developed by the Oceanographic Systems Laboratory at Woods Hole Oceanographic Institute. In 2001, REMUS AUVs entered commercial production and they are currently sold by Hydroid, Inc. REMUS is used for a variety of applications which include environmental sensing, harbor security, and mine countermeasure operations. The vehicle operates with a laptop computer. Launching and recovery operations are simplified due to the vehicle's compact size and light weight. As seen in Figure 2, it is a small portable system that is 7.5" (19 cm) in diameter, 63" (160 cm) long, and weighs 80 pounds (37 kg). As defined by the vehicle classifications in the US Navy UUV Master Plan the REMUS AUV is considered a man-portable vehicle. As a package, REMUS incorporates a wide range of onboard sensors and includes an upgradeable payload for the addition of unique sensor packages. All of these factors make REMUS an attractive platform for US Navy applications. Furthermore, research

tailored to the REMUS platform has the distinct advantage of applying directly to a vehicle already in production and presently deployed by the US Navy. The U.S. Navy currently uses variants of the REMUS AUV to assist the Naval Special Clearance Teams to locate mines.

Figure 2. REMUS AUV (From Hydroid Inc., 2007)

REMUS can be configured with many different types of sensors such as: side scan sonar, an ADCP, inertial navigation system, and acoustic modem. The navigation system includes a compass, the above-mentioned ADCP to provide speed over ground when ground lock is available, and an acoustic LBL system to correct accumulated dead reckoning errors. REMUS simultaneously senses its depth under the surface of the water and uses it's Teledyne Technologies Inc. RD Instruments (RDI) Workhorse Navigator based ADCP DVL sonar to detect its altitude above the ocean floor. The ADCP DVL is also used to calculate the ground-referenced or water-referenced vehicle velocity. Currently side scan sonar is employed to detect objects on or near the sea floor.

PHYSICAL/FUNCTIONAL AREA	CHARACTERISTIC
Vehicle Diameter	19 cm
Vehicle Length	160 cm
Weight in Air	37 kg (<80 lbs.)
Trim Weight in Air	1 kg
Maximum Operating Depth	100 meters
Energy	1kw-hr internally rechargeable Lithium ion
Endurance	22 hours at optimum speed of 1.5m/s (3 knots). 8 hours at 2.6m/s (5 knots)
Propulsion	Direct dive DC brushless motor to open three bladed propeller
Velocity Range	0.25 to 2.8 m/s variable over range
Control	2 coupled yaw and pitch fins
On/Off	Magnetic switch
External Hook-up	Two pin combined Ethernet, vehicle power and battery charging; 4pin serial connector
Navigation	Long base line; Ultra short base line; Doppler assisted dead reckon; (Optional: GPS)
Transponders	20-30 kHz operating frequency range
Tracking	Emergency transponder, mission abort, and ORE Trackpoint compatible
Sensors Doppler Velocity Log	RDI 1.2 MHz up/down looking
Side Scan Sonar	600 or 900 kHz MSTL AUV model

Table 1. REMUS Specifications (From Hydroid Inc., 2007)

The Naval Postgraduate School REMUS AUV is equipped with a forward looking sonar (FLS) that is normally used for detection of objects on the bottom and within the water column as well as for obstacle avoidance. This FLS is a low-power, high-resolution Blue View Technologies Blazed Array active sonar that operates at 450

kHz. The FLS provides a 45 degree field of view and has an effective range of 450 feet, or 137.2 meters. The range resolution of the FLS is adjustable, and the two Blazed Array transducers are also reconfigurable. The Blazed Array sonar is discussed further.

D. THESIS SCOPE AND STRUCTURE

The goal of this thesis is to utilize sequential Blazed Array sonar images to accurately estimate forward and lateral velocities as well as the heading rate of an AUV. Numerous component problems must be addressed to achieve that goal. The sonar images are a matrix in Cartesian coordinates comprising of pixels whose values (16 bit, 0 to 65535) represent intensities of the return strength of the forward looking sonar. A template matching, or image correlation, algorithm is presented, where the previous sonar image is modified to simulate motion of the AUV. Euclidean transformations using a combination of translation and rotation will simulate motion of the vehicle in the image. The correlation coefficient is calculated comparing the images. A search is performed and the transformation which maximizes the correlation coefficient is converted to estimates in the forward velocity, lateral velocity, and heading rate through motion analysis. The estimated velocities and heading rates is compared to the ADCP DVL measured velocities and the compass measured heading rates. The velocity estimates could be used as inputs into the AUVs control algorithms and steering model, replacing the inputs from current velocity measurement techniques. The effect on the navigation performance of the AUV steering model can be determined under these conditions.

Figure 3. REMUS AUV (From UUV Master Plan, 2004)

Chapter II will present the theory of active sonar, covering specifically the operation of the ADCP as it utilizes Doppler Effects to measures velocity, and the operation of the Blazed Array transducers. Chapter III will provide the details of the image processing and computer vision techniques applied to estimate the velocities of the AUV. Due to the interdisciplinary nature of this thesis, previous related work is discussed in the chapter introductions and the applicable sections. Chapter IV will detail the steering model for the REMUS AUV. Chapter V will present the vehicle simulation in detail and the simulation results. Finally, Chapter VI provides thesis conclusions and recommendations for future work. The supporting code utilized in this work is retained in the appendices to this thesis.

II. ACTIVE SONAR

A. INTRODUCTION

Active sonar is the use of a transmitted acoustic signal to navigate and locate features. The physical propagation of that signal in water can be modeled and accurate ranges and bearings to features can be determined from the returned acoustic signal, which is also called an echo. The active sonar process is a method of echo-locating features in the underwater environment. A transducer both produces an acoustic pulse, or 'ping', and listens for the reflected return signal. Range as well as bearing to a feature or object can be determined from the return signal. (Waite, 2002) The time measured from the transmission of the acoustic signal (t) and the speed of sound in the water (c) is used to calculate the range to the feature that resulted in the return signal.

$$\text{Range} = ct / 2 \tag{1}$$

The reflected signal which is detected contains information, besides location, about the feature. The intensity and size of the return signal can be used to aid in the identification of specific features. Images are created from the returned acoustic signal. Based upon the time that return signal is measured and the speed of sound propagating through the water as well as the bearing that the signal is received, the location of features can be accurately displayed within the image. (Figure 4) However due to the process of the propagation of sound in the water there is significant noise associated with the acoustic signal. This noise will affect the return signal measured by the transducer. The inherently noisy nature associated with sonar is much greater than the noise that would be associated with optical images (Cuschieri, 1998). Motion analysis becomes more challenging with the noisy nature of the sonar images.

The continued development of active sonar has resulted in a variety of systems; however the concept of utilizing the propagation of sound through the water has remained constant. Many types of acoustic signals have been designed; continuous wave pulses and frequency modulation pulses are examples. Utilizing digital signal processing the signals can be manipulated to form specific beam patterns. Various frequencies are

used based upon their propagation performance in the undersea environment. Sonar can also be used to measure the Doppler shift of contacts. This frequency shift between the transmitted and received signals is a result of the relative motion of the contact with respect to the transmitting platform. The RDI ADCP utilizes the Doppler shift of the active signals to measure the velocity of the REMUS AUV. The process by with the change in frequency between the transmitted and received signals is converted into a measurement of the vehicle velocity is discussed further.

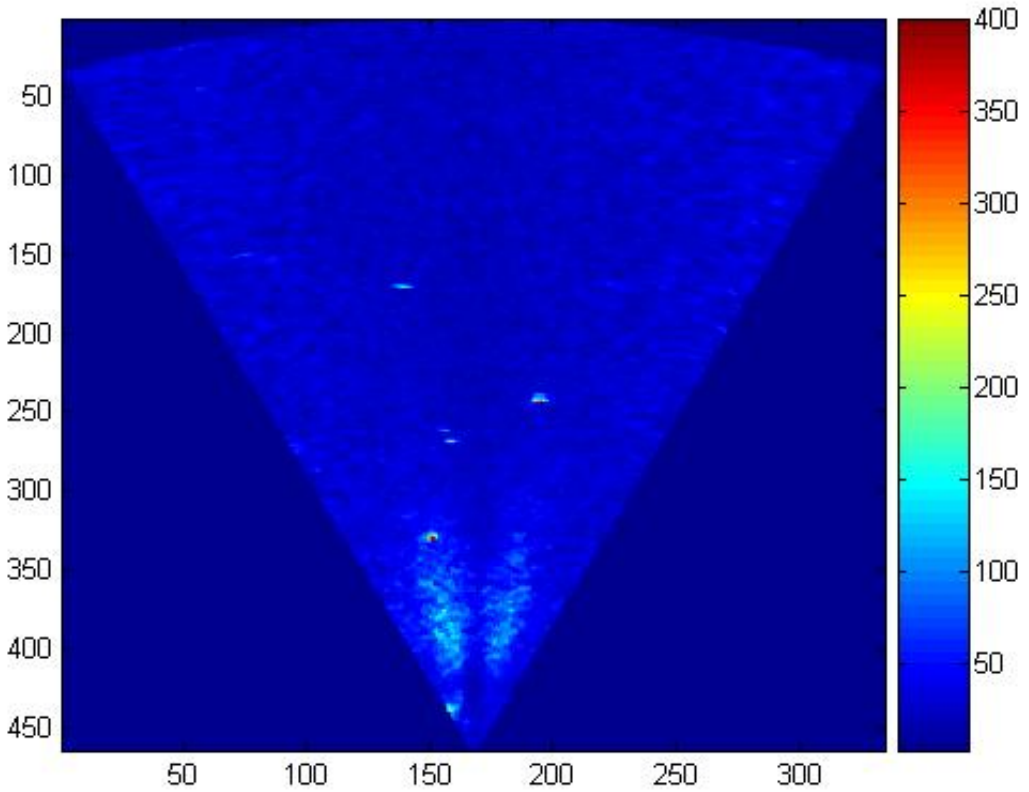

Figure 4. Blazed Array FLS Image of Multiple Features REMUS AUV 012506.

B. ACTIVE SONAR EQUATIONS

Active sonar systems transmit a pulse of sound and then listen for return echoes. The sonar equation accounts for how intense the sound source is (source level), sound spreading and attenuation as the sound pulse travels from the sonar to the target

(transmission loss), the amount of sound reflected back toward the sonar by the target (target strength), sound spreading and attenuation as the reflected pulse travels back to the receiver (transmission loss), the background noise at the receiver (noise level), and the receiver characteristics (array gain). The terms in the sonar equation are all in decibels, and they are added together forming the active sonar equation.

The sonar transmits a signal with a source level SL, given in underwater decibels referenced one meter from the source. The sound becomes weaker as it travels toward the target, due to spreading and absorption. The total reduction in signal intensity is called the transmission loss TL. The sound intensity at the target is then (SL -TL). Only part of the sound that hits the feature is reflected back toward the sonar source. The intensity of the echo one meter from the target relative to the intensity of the sound hitting the target is called the target strength TS. (Waite, 2002) The echo one meter from the target essentially looks like the signal from a source with a source level of:

$$\text{Echo intensity (decibels)} = (SL - TL) + TS \tag{2}$$

As the reflected signal travels back to the sonar system, the signal intensity is again reduced by the transmission loss TL. The intensity of the returned signal or echo at the receiver is then:

$$\text{Returned signal intensity (decibels)} = (SL - TL) + TS - TL \tag{3}$$

which can be simplified to:

$$\text{Returned signal intensity (decibels)} = SL - 2TL + TS \tag{4}$$

If the noise level at the receiver is NL decibels, then the ratio of the signal level to the noise level at the receiver, called the signal-to-noise ratio (SNR), is:

$$\text{SNR (decibels)} = SL - 2TL + TS - NL \tag{5}$$

As can be seen from the developed active sonar equation, the intensity of the return can depend on many factors. The propagation of sound in the water, angle of incidence, range, feature hardness, water and environmental conditions can all affect the intensity of the active sonar return signal.

C. DOPPLER VELOCITY MEASUREMENTS

The Doppler Effect is the change in frequency and wavelength of a wave that is perceived by an observer moving relative to the source of the waves. For waves, such as acoustic waves, propagating though the ocean, the velocity of the observer and of the source is reckoned relative to the medium in which the waves are transmitted. The total Doppler Effect may therefore result from both the motion of the source and the motion of the observer. Doppler sensors have been used for several years to aid in navigation. The Doppler sensors calculate the AUV velocity relative to the sea floor or the water column (they also measure water currents). The sensors transmit a high frequency narrow beam. Due to the motion of the vehicle the frequency of the returned signal is slightly different. The shift in frequency is then used to calculate the velocity of the vehicle.

The REMUS AUV has a 1200 kHz Teledyne RDI Workhorse Navigator ADCP DVL. This ADCP is a 4 beam sensor in a Janus configuration (facing opposite directions) with a 60 degree depression angle. The sensor is both upward (Figure 5) and downward looking.

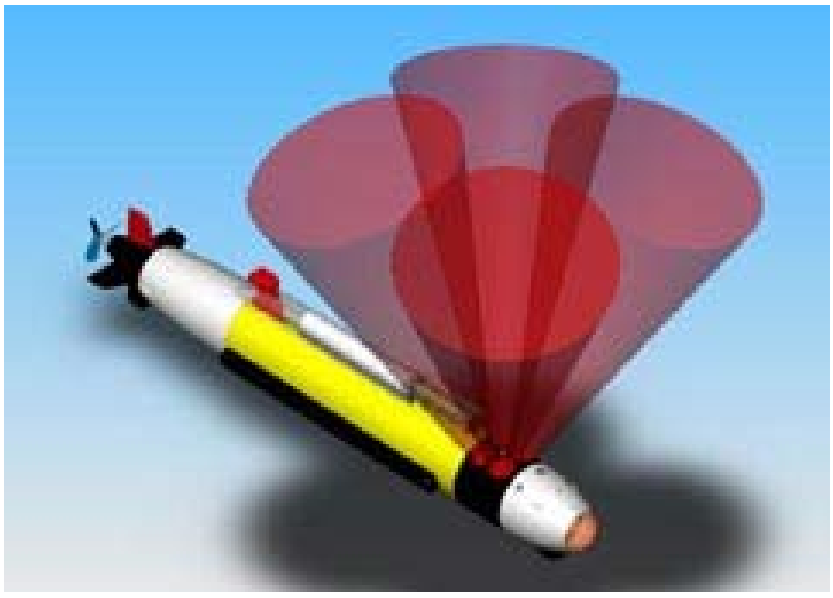

Figure 5. REMUS ADCP Upward Beams (From NOAA's Marine Navigation 2007)

The ADCP measures the two way Doppler shift. As the acoustic signal travels from the transducer to the bottom or surface, there is a Doppler shift due to the moving transmitter, which is the source, and a stationary bottom. Once the acoustic signal reaches the surface or bottom it is reflected and scattered. Some of the acoustic signal travels back to the transducer; therefore the Doppler shift is due to the stationary bottom, which is now the source, and the moving transducer. The two Doppler shifts are not equal. As derived by Jorgensen the velocity of the vehicle (v) the frequency of the transmitted signal received at the bottom is

$$f_b = \frac{f_o}{1 - \frac{v}{c}\cos(A)} \tag{6}$$

Where f_o is the frequency of the transmitted signal, v is the vehicle speed, c is the speed of sound, and A is the transmission depression angle from the horizontal. The acoustic signal is then scattered back to the AUVs transducer from the bottom. The frequency shift due to the Doppler, from the scatters to the transducer is the received frequency.

$$f_r = \frac{f_o\left(1 + \frac{v}{c}\cos(A)\right)}{1 - \frac{v}{c}\cos(A)} \tag{7}$$

The delta Doppler frequency is the difference between the transmitted frequency and the received frequency. This would be the delta Doppler frequency for a single acoustic beam.

$$f_d = f_r - f_o = \frac{2f_o\frac{v}{c}\cos(A)}{1 - \frac{v}{c}\cos(A)} \tag{8}$$

Since the denominator is approximately unity the equation can be reduced to the following

$$f_d \approx 2 f_o \frac{v}{c} \cos(A) \tag{9}$$

Using the Janus configuration, the acoustic beams which are on opposite sides can be used to reduce the errors associated with this equation. Using the delta Doppler frequency from a fore and aft beam, then the equation becomes

$$\Delta f = \frac{\left(f_{d\,fore} - f_{d\,aft} \right)}{2} \tag{10}$$

Which can be rewritten as

$$\Delta f = \frac{2 f_o \frac{v}{c} \cos(A)}{1 - \frac{v^2}{c^2} \cos^2(A)} \tag{11}$$

Since the second term of the denominator is negligible, the equation can be reduced to

$$\Delta f = 2 f_o \frac{v}{c} \cos(A) \tag{12}$$

However, the depression angle of the transducers is 60 degrees; therefore the equation can be simplified (Jorgensen, 1993)

$$v = \Delta f \lambda \tag{13}$$

Since the ADCP utilizes the same principles as other active sonar systems the same errors are associated with the systems. The ADCP utilizes a higher frequency than the FLS and side scan sonar, however the principles associated with the propagation of sound through the water remains unchanged. The advertised long-term accuracy of the Teledyne RDI Workhorse Navigator ADCP DVL is 0.2% or 0.1 centimeters per second.

D. NOISE ASSOCIATED WITH SONAR SYSTEMS

The inherently noisy nature associated with sonar images (Cuschieri, 1998) makes the process of motion analysis challenging. There are several sources of noise associated

with active sonar. Sonar systems usually need a level of SNR to determine if a contact is detectable. Noise can come from thermal/electrical noise, ambient noise, vessel noise, and reverberation. Thermal/electrical noise is produced by the electrical system associated with the sonar. Any resistance within the sonar system is a source of thermal noise. (Waite, 2002) Sonar designers take this noise into consideration, and sonar systems are designed to keep this noise at a relatively low level. Ambient noise, which is also referred to as background noise, includes all of the noise in the ocean. Processes such as wind and rain can significantly increase the level of ambient noise by increasing the sea state. The formation and collapse of small air bubbles is a noisy evolution. Shipping in harbors and transient lanes increases the ambient noise. Also a variety of marine life can increase the ambient noise, for example marine mammals and shrimp can produce high levels of noise. Vessel noise is the noise created by the vehicle itself. This would include flow noise and the noise associated with propulsion and additional machinery. For a small battery powered AUV operated at relatively slow speeds these vessel noise sources would be small.

Reverberation is a considerably more significant source of noise. Reverberation is a result of the active acoustic signal being scattered. Scattering can be caused by marine life, inanimate matter suspended in the water column, and even the inhomogeneous structure of the water column itself. Significant scattering can also be caused by the ocean surface and the sea bed. Some of the scattered signal is directed back to the transducer. This component is called backscatter, and this energy is reverberation. Many of the current AUVs operate in the littoral environment. Littorals are near-shore areas, which are shallow water environments. The shallow waters results in high reverberation from both the surface of the ocean and the sea bed. These near shore and harbor areas also have higher concentration of suspended inorganic material. This increased concentration of matter in the water column can also increase the level of reverberation.

The active acoustic signal is affected by a variety of factors. Environmental conditions such as the concentration of marine life and suspended inorganic material in the water can affect the noise level in the acoustic return signal. As the concentration of

particulate material in the water increases then the amount of reverberation also increases. Therefore the SNR would decrease under these conditions.

E. BLAZED ARRAY TRANSDUCERS

The Naval Postgraduate School REMUS AUV is equipped with a forward looking sonar (Figures 6 and 7). This sonar is typically used for detection of objects on the bottom and within the water column, as a gap-filler for the side scan sonar, well as for obstacle avoidance. This FLS is a low-power, high-resolution Blue View Technologies Blazed Array active sonar that operates at 450 kHz. The FLS provides a 46 degree field of view in the imaging plane and 15 degree field of view perpendicular to the imaging plane (Figure 8). The FLS has an effective range of 450 feet, or 137.2 meters. The range resolution of the FLS is adjustable, and the two Blazed Array transducers are also reconfigurable.

Figure 6. Naval Postgraduate School REMUS AUV with BLUEVIEW FLS

Figure 7. Naval Postgraduate School REMUS AUV with BLUEVIEW FLS. Note the nose cone is removed, and the transducers are in the horizontal configuration.

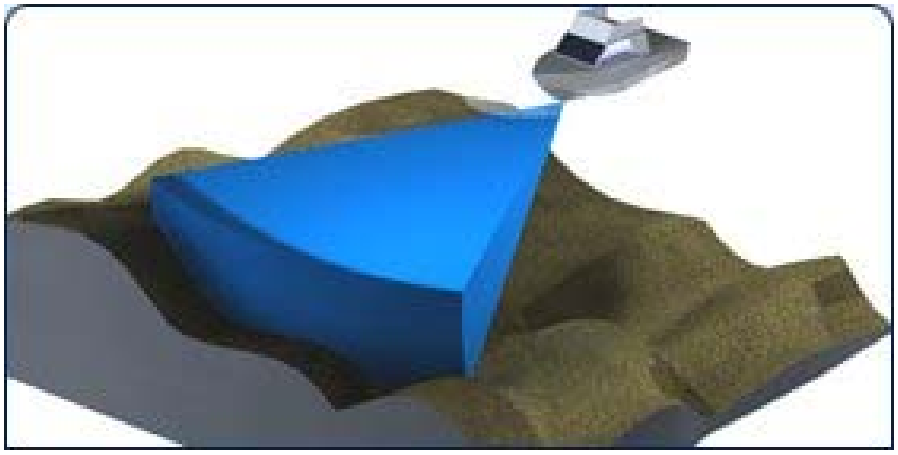

Figure 8. Field of View of a Blazed Array transducer (From BlueView Technologies Inc., 2007)

P450-15-RS BLUEVIEW FLS CHARACTERISTICS	
Max Range	450 ft
Update Rate	Up to 10 Hz
Swath Width	45 degrees
Beam Width	1.0degree x 15 degrees
ELECTRICAL	
Power	19-35 volts DC @ 25 watts
Communications	Ethernet
Communications Settings	IP Address 192.168.1.100
MECHANICAL	
Depth Rating	100 m
Weight in air w/o PC104	12.2 lbs
Weight in water w/o PC105	0.3 lbs (salt) / 0.6 lbs (fresh)
Dimensions Length w/ Nose Cone	11.4 in
Dimensions Width	7.5 in
ACOUSTIC	
Operating Frequency	300-600 kHz
Number of Beams	20

Table 2. Blue View FLS Specifications (From Blue View Inc., 2007)

Scanning type sonars are common and work by mechanically rotating a single acoustic beam over the imaging area. This method is less accurate when used from a moving platform, such as an AUV. The Blazed Array transducers produce many small acoustic beams simultaneously. Blazed Array transducers generate an acoustic beam with a series of frequencies ranging from 300 kHz to 600 kHz, where each frequency is radiated at a specific characteristic angle (Figure 9). Each beam incrementally increases 15 kHz, for a total of 20 individual beams which are transmitted with each ping. In essence, this process is frequency steered acoustic beamforming. Multiple independent beams can be simultaneously formed from a single hardware channel, which allows for

smaller sonar designs which are cheaper and require less power. These small sonars are well suited for AUVs (Thompson, 2001).

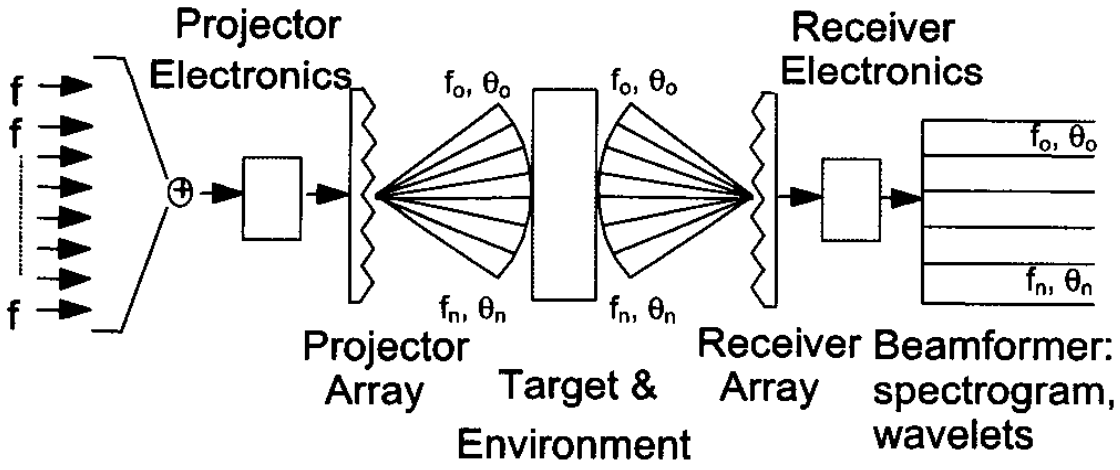

Figure 9. Illustration of Blazed Array Beamforming (From Thompson, 2001)

The overall concept of active sonar still applies with the Blazed Array transducers. The broadband signal is generated, which interacts with the environment and targets within it. Backscattered signal is then received by the transducers, and the image is generated. The significant difference with Blazed Array transducers is that angular imaging information is embedded into the transmitted broadband signal through the frequency domain.

Imaging sonar results in a two dimensional image projection of the three dimensional environment. The center of the projection is the face of the Blazed Array transducers, which is a finite distance from the projection plane. The two dimensional projection is created from the acoustic reflections due to features within the sonar's field of view. Another important characteristic of the Blazed Array sonar images are that they are generated from two individual staves. Mismatched staves could result in poorer results due to the poorer acoustic performance of the sonar.

F. NEAR-FIELD EFFECTS

During the initial model simulation it was observed that some of the results estimated that the forward velocity of the vehicle was zero. An evaluation of these results and an examination of the sonar displays resulted in images that had no significant features within it, but had near-field effects. These effects, as seen in Figure 10, are intense returns that are directly in front of the two staves of the Blazed Array. These near-field intensities returns are strong signals and are detected forward of the sonar in the two sequential images. Therefore, when the image matching is conducted by maximizing the correlation coefficient the vehicle velocity in the forward direction is approximated to be zero. The intense near-field effects extended up to include more than a quarter of the sonar field of view. Further pier-side experimentation which included only the sonar and not the AUV do not exhibit these near-field effects. This would seem to suggest that the effect could be caused by noise associated with the vehicle. During the experimentation the sonar was also limited in the range of motion. There was no forward motion, and therefore no flow noise. There was no ADCP or side scan sonar, therefore the was no additional acoustic noise being transmitted into the ocean

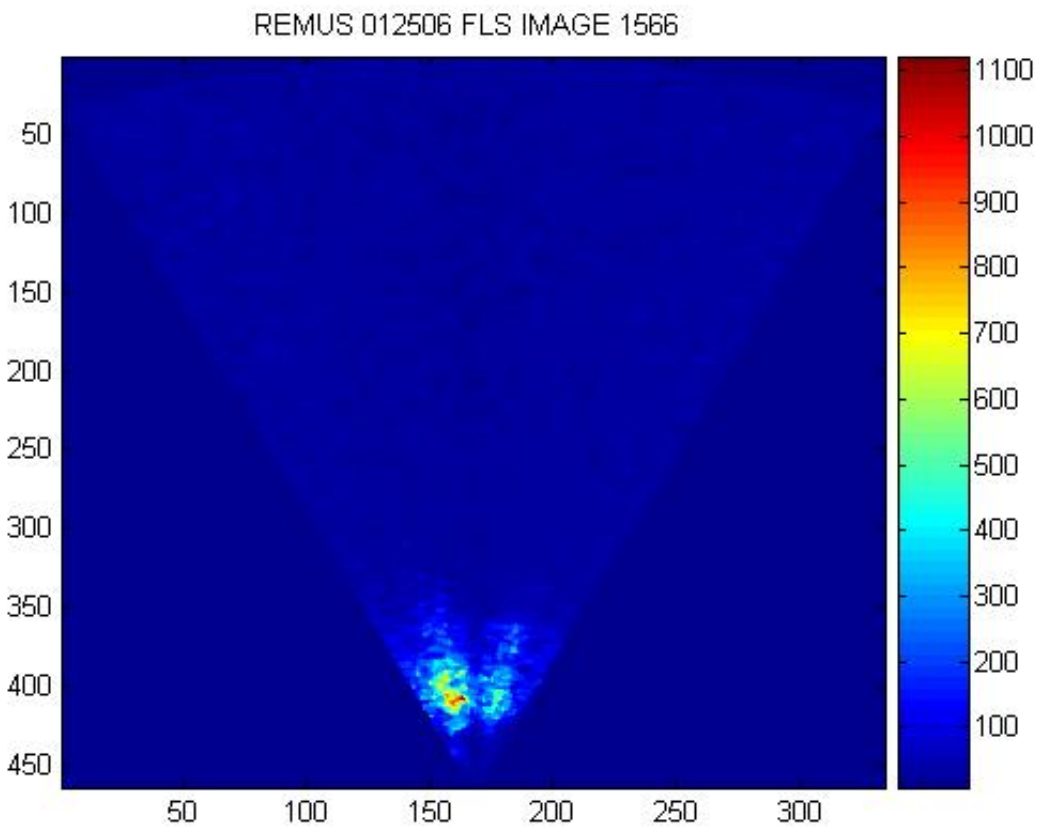

Figure 10. FLS Image with Dominate Near-Field Effects REMUS AUV 012506.

Pitch and roll of the AUV were examined to determine if these near-field effects were caused by reverberation due to the surface of the ocean or the sea floor. There appeared to be no correlation to the pitch and roll of the vehicle to these effects. Also evaluated were the conductivity, temperature and depth measurements from the mission to evaluate if this effect could be a result of environmental conditions such as increased suspended particulate or inhomogeneous conditions within the water column. However there was no indication that any environmental conditions were causing the near-field effects. The level of the transmitted signal was examined to determine if insonofication could be a result from excessive peak power, produced by the transducer. The transmitted acoustic signal was not sufficient enough to cause the formation of bubbles directly in front of the staves. BlueView Technologies, Inc. confirmed that peak power

associated with the source level of the FLS is less than the peak power necessary to result in bubble formation.

From a discussion with personnel from BlueView Technologies, Inc. the near-field effect is possibly a result of the frequency beamforming. In an effort to produce the most appropriate beams from the two staves, signal processing is used to manipulate the transmitted signal. There are many benefits and costs associated with manipulating the signal. Range resolution, beam width, and side lobe strength and numbers are all affected by the beamforming.

Additional experimentation with the FLS detached from the REMUS AUV resulted in images which did not present the near-field effects. This suggests that the near-field effects are a result of some aspect of the AUV and not the actual sonar system. The source of the near-field effects requires additional research and will be discussed further in recommendations for future work.

III. COMPUTER VISION

A. INTRODUCTION

The optics and optical processing associated with the human eye is complex. There are significant amounts of processing accomplished, at both low level and high level. People can rely on inference and assumptions, while computing devices must 'see' by examining individual pixels of images, processing them and attempting to develop conclusions with the assistance of knowledge bases and features such as pattern recognition engines. Computers rely upon sensors which are not equivalent to the human optics. Sensors such as optical and infrared cameras as well as sonar and radar provide digital signals which the computers must analyze. Computers do not 'see' in the same way that human beings process optical signals. A benefit of computer vision systems is that they are capable of processing images consistently. However, computer-based image processing systems are typically designed to perform a single, repetitive task, and despite significant improvements in the field, no computer vision system can yet match some capabilities of human vision.

Although some computer vision algorithms have been developed to mimic human visual perception, a number of unique processing methods have been developed to process images and identify relevant image features in an effective and consistent manner. Computer vision can be considered the method to describe a scene or extract useful information from the scene. Machine vision is the application of computer vision to industry and manufacturing. The goal of machine vision is to recover useful information from the images and then apply that information. Machine vision most often uses digital input/output devices and computer networks to control other equipment such as robotic arms, or control surfaces on an AUV. Machine vision is a subfield of engineering that encompasses computer science, optics, mechanical engineering, and industrial automation.

The Navy UUV Master Plan identifies several of the areas where research and development continues to be required. The development of autonomy and control as well

as sensors and sensor processing are areas requiring more research and development. Machine vision has a role in the further developing the autonomy of unmanned vehicles. This thesis applies computer vision techniques to the sonar images.

Machine vision methods have been applied extensively to images from cameras for use in navigation and determining unmanned vehicle positioning. Unmanned aerial vehicles and space vehicles have used inputs from cameras to aide in navigation and control (Roumeliotis, 2002). Some AUVs have also used cameras for navigation and control (Kalyan, 2004). The potential of using optical cameras for navigation in the undersea environment is limited, since light does not travel the distances necessary to make this a practical method in the undersea environment. However the machine vision methods have not been applied as extensively to sonar images. Sonar imagery traditionally has been used to detect, localize, track, and identify targets of interest.

B. MOTION ANALYSIS

For many years now images from cameras have been used to determine the motion of objects within the scenes. Motion analysis is frequently based on a small number of sequential images. Typically, points of interest are identified, analyzed and velocity vectors are created from the pairs of points, or features, in the sequential images. The points of interests are usually identified using a feature detector, which is discussed further.

Optical flow is a concept for estimating the direction and speed of instantaneous motion of intensity points within a sequence of visual images. Many researchers have used optical flow methods to determine the velocities of objects within a sequence of camera images. Appling this idea to the sonar images has resulted in the development of acoustic flow, which involves the estimation of the range and azimuth rates of the features within the sonar images (Cuschieri, 1998). Their experimentation was conducted using significant features such as a sunken barge; however it was accomplished using only sonar images, where previous work employed cameras. The complex noise associated with the sonar images was apparent when compared to the optical imagery from cameras. This noise was identified as a challenging problem to the

acoustic flow method, and a theoretical alternative identified by Cuschieri and Negahdaripour was to estimate motion directly from the intensity variations within the sonar imagery, which is what is proposed within this thesis. There are several approaches for estimating velocity; in this thesis the model will employ a correlation coefficient based matching procedure followed by motion analysis. Motion analysis is done utilizing the subtle patterns within the sonar image due to the variability in the ocean bottom. The key attribute to the method of using the template matching techniques and the correlation coefficient is that velocity estimates can be made even when there are no intense features within the sonar image to track.

National Aeronautics and Space Administration is utilizing vision to help aid in motion estimation for vehicles to land on distant bodies (Roumeliotis, 2002). Their work involves cameras, and tracking distinct features between consecutive images. Then they use a rigid transformation, and a cost function to estimate the motion, where they minimize the cost function to optimize the motion estimates. However, this work was done using a high resolution camera. Similar work was done in an underwater environment (Kalyan, 2004). Here they used a high resolution monocular camera to estimate the motion. They needed to conduct a high degree of filtering to the images to remove scattering by the suspended particulate (Kalyan, 2004) A corner detector was then used to extract a large number of features, and a correspondence method which compares all corners detected within the two images. The corresponding points are then used to estimate the homography between the two frames. The homography is the relationship between the two images, where any point in one image corresponds to one and only one point in the other, and vice versa. The homography contains the rotation and translation, which is then converted to an estimate of motion. Additional work involved optical triangulation using the reflection of lasers off the bottom (Caccia, 2002), where the reflections were identified through their intensity. The characteristic problems associated with underwater vision, such as suspended particulate, limited range, non-uniform lighting, and the unstructured environment were identified.

Previous work which involved the Blazed Array sonar utilized the transducers in the vertical configuration. Using image processing the ocean floor is identified within the

sonar image. A Hough Transform was used to determine the height of obstacles above the ocean floor. The AUV used this as an input for reactive obstacle avoidance (Horner, Healey, Kragelund, 2005) within the vehicle's autopilot controller to provide greater autonomy. Optical flow techniques applied to the sonar images were identified as an additional method for obstacle avoidance.

The process within this thesis involves two sequential sonar images which is analyzed. These FLS images are created from the intensity returns of the active acoustic signal. The image is a matrix in Cartesian coordinates where individual pixels value represents the intensity of the return signal. A critical assumption is that objects within the image are stationary and that the motion within the image is due solely to the vehicle. This assumption would break down if there were significant concentrations of suspended particulate within the water column. The assumption could also break down if the vehicles operating area had significant kelp or other types of seaweed growing on the bottom that could potentially sway with the currents. A search is conducted utilizing a Euclidean transformation which is then performed on the image. The transformation is applied where the rotation is directly related to the heading rate, and the translation is directly related to the forward and lateral velocities. The best match between the two sequential images is then determined by calculating the correlation coefficient of the two image matrices. The transformation associated with the best match is then used to estimate the velocity in the forward and lateral directions as well as the heading rate.

C. FEATURE IDENTIFICATION

Feature detection has been an area where there has been much development. There are many types of feature detectors that have been used successfully within image processing. Depending upon the features that an operator expects to find and the features that an operator wants to track within the program, there are many detectors from which can be chosen. Feature detection is the process by which a program examines a particular image and finds points of interest that can be easily found with the next image. Examples of proven feature detectors are edge detectors, corner detectors, and line detectors. In an image, these features are identified by the abrupt changes in intensity or brightness

between the pixels (Sonka, 1995). A corner detector was then used to extract a large number of features from underwater camera images in previous work (Kalyan, 2004). Most feature detectors have been developed for visual camera images, some of these were considered for use in identifying features with the sonar image. The line detector was considered due to the sonar's characteristically good range resolution.

When considering a feature detector for the sonar image, researchers first must understand the actual sonar image itself. A sonar image is created from the intensity of the return signal of the detections, vice a combination of colors that create a camera image. The intensity itself is an actuality, a detection of a feature. Therefore by remaining within the image-space, it is not necessary to utilize a separate and distinct feature detector. Within this thesis the images are compared using the calculated correlation coefficient, which performs the pixel comparison within the image space. The removal of this process provides a reduction in processing time and would reduce the processing power necessary to perform the operation. In this thesis it is not necessary to track individual features, however if this process was used in cooperation with a feature-based navigation program it would be necessary to identify and track individual features between the images. Feature-based navigation programs detect the features and track them to create a map of the environment with which the AUV then can use to navigate. This difference, utilizing dense image matching, results in higher correlation than sparse feature location matching. However, determining the correlation with dense image matching requires more processing time.

D. EUCLIDEAN TRANSFORMATION

Sequential images from the forward-looking sonar are taken while the vehicle is traveling through the undersea environment. The goal of this thesis is to compare the images to identify and estimate that motion in the forward and lateral direction and heading rate. The assumption has been previously stated that all of the features detected by the sonar are stationary bottom features. Therefore the apparent motion by the features is due solely to the motion of the vehicle. To compare the sonar images the estimated motion is applied to the previous image. This is done using an image

processing technique called an affine transformation. There are several types of image transformations, in this thesis a combination of rotation and translation is applied to simulate the estimated motion of the vehicle.

In essence, each feature, as identified through intensity, within the first image is matched to the same intensity within the second image. When this matching is conducted the current image is held stationary while the previous image is adjusted using a transformation to find the correct dr/dt and dθ/dt that maximizes the correlation coefficient. The transformation that maximizes the correlation coefficient will then be converted to estimates of the motion between the two sonar images. These processes of rotation and translation are applied to the images within the image space.

To simulate motion between the previous sonar image and the currently observed sonar image the rotation and translation applied must rigid motion with no scaling or distortion. The transformation method must preserve the size of the image and the lengths between the features; therefore there is no distortion within the image. The rigid-body model requires that the real world Euclidean distance between any two pixels coordinate locations to remain unchanged by the transformation (Jain, 1995). The Euclidean transformation, also referred to as rigid-body transformation, is used within this thesis since it prevents distortion within the images.

Euclidean transformation normally utilizes a rotation matrix and a translation matrix. In Figure 11 $T_{i,j}$ represents translation vector between images `j' and `i' and $\alpha_{i,j}$ rotation angle of image j in image i coordinate system (Sonka, 1995). Euclidean transformations preserve length and angle, so the shape of an object within the image does not change only the position and orientation of the object changes (Figure 11).

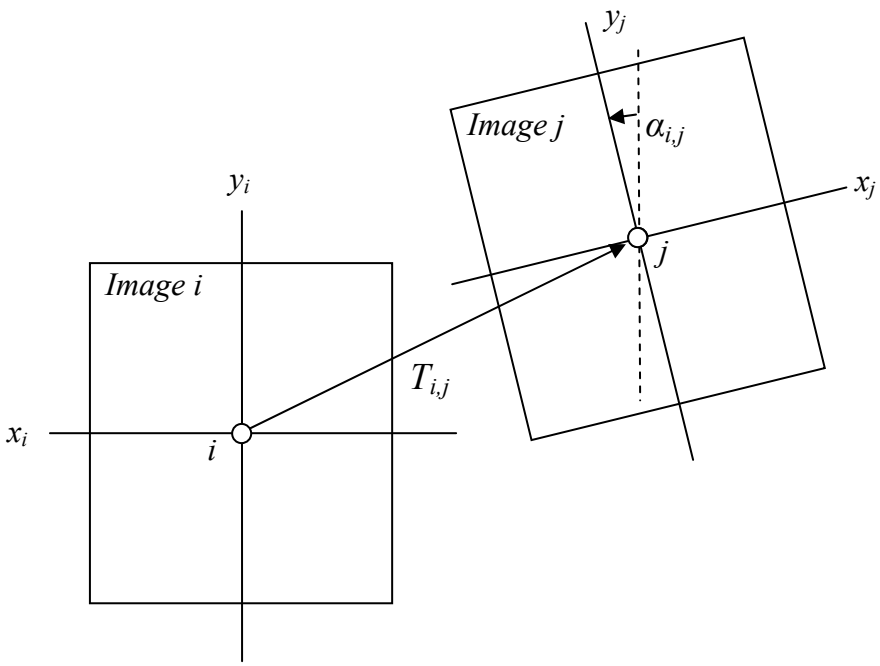

Figure 11. Rigid Transformation Model of Image Displacement

Euclidean transformations can be decomposed into two operations, first translation and then rotation, to simulate forward and lateral velocities and then heading rate. Any Euclidean transformation can be represented as a matrix of appropriate size. For example:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathrm{R} \begin{bmatrix} x \\ y \end{bmatrix} + \mathrm{T} \tag{14}$$

Where R is a rotation matrix and T is a translation vector. The two dimensional Euclidean rotation matrix using homogenous coordinates is:

$$\mathrm{R}(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{15}$$

Once the rotation and translation are applied to the sonar images the correlation coefficient can be calculated. The correlation coefficient conducts a pixel-by-pixel

comparison of the two images. The presence of pixels outside the field of view would provide unrealistic results when the correlation coefficient is calculated and therefore would result in inaccurate estimates of velocity. These overlapping areas which are outside the combined field of view of the two sonar positions are removed to ensure an accurately calculated correlation coefficient.

E. CORRELATION COEFFICIENT BASED MATCHING

Matching has been used to determine image positions where known objects and specific patterns are located (Sonka, 1995). In a stereoscopic scene where more than one image of a scene is taken from different locations, matching can be used to determine scene properties. Matching has been used in the undersea environment to create mosaics of the ocean bottom (Fleischer, 1998). In this thesis the previous sonar image is the pattern against which the current sonar image is compared. The patterns that is compared are the slight variations in the intensity of the sonar returns. Even without strong returns from features, there is slight variation within the image due to the characteristics of the bottom. Ripples in the sand, for example, can give slightly different return intensities due to the variation associated with the acoustic signal scattering off of the ripple. The correlation-based approach requires the assumption that the relative local intensities within the image remain constant.

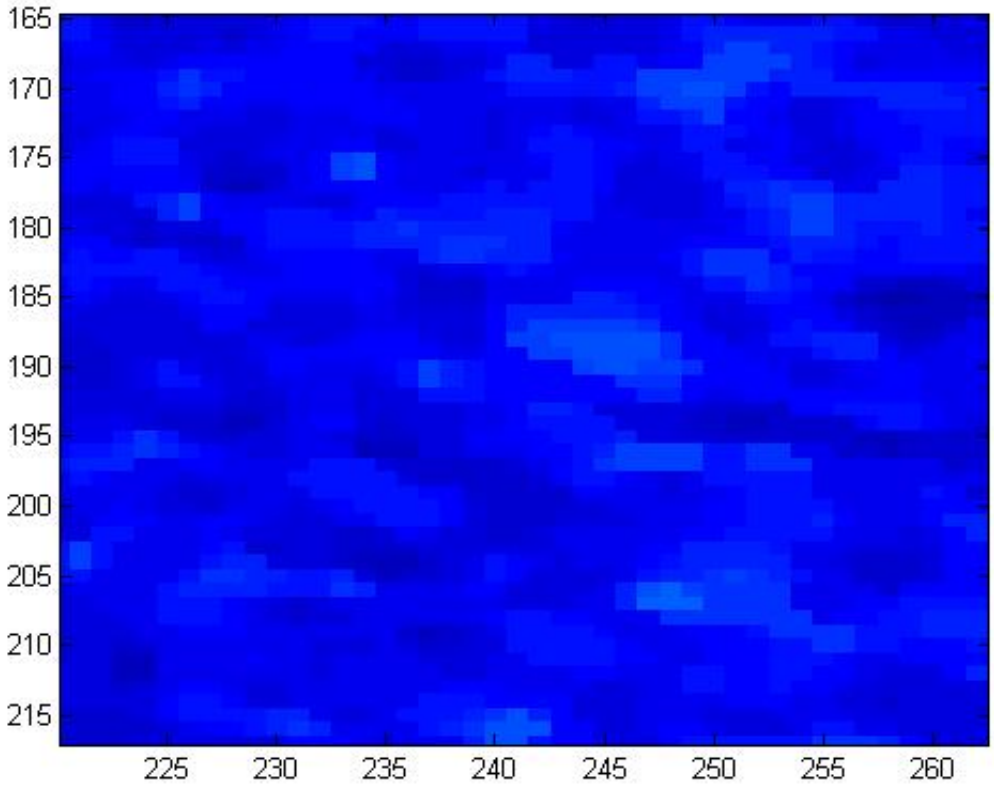

Figure 12. Example of a pattern in FLS image from REMUS AUV 012506

In essence, the pattern within the first image is matched to the same pattern within the second image. Using the sonar images as matrices of intensities the correlation coefficient can be calculated. It is not necessary to use a feature detector to identify and then track specific points between the images. The use of the entire image actually results in higher correlation than sparse feature location matching. Utilizing a search and maximizing the correlation coefficient between the two sequential images, the optimal rotation and translation associated with the two images can be determined. The translation is then converted to velocities estimates in both the forward and lateral directions and the rotation is converted to an estimate of heading rate.

Sonar images were compared and evaluated to determine the intensities that were being observed and used to conduct the matching and subsequent velocity estimates. The near-field effects (Figure 10) and the mine-like features (Figure 4) had strong intensities

as can be seen by the pixel values. Mine-like features were placed on the ocean bottom to simulate a mine field. The features intensity varied between sequential images but was strong, easily exceeding values of 700. The average value of the intensity in the far-field sonar field of view, not including the features, varied from 30 to 50. It is the patterns generated within the sonar images at these low intensities that result in the velocity estimates (Figure 12).

The correlation coefficient is a measure of how well the patterns within the two images "match". Matching has been used to identify a portion, specific object, or pattern, within a larger image. This matching technique is applied to dynamic images. Matching is rarely perfect since the pattern is usually corrupted by various sources of noise and geometric distortion therefore an absolute match is not possible (Fleischer, 1998). The value of the correlation coefficient ranges from negative 1.0 to positive 1.0, where a perfect match would be exactly 1.0. The prefect match would exist when every single pixel within one image is the identical value in the second image. With the introduction of noise, a precise match is impossible. However the maximum match, as measured by the correlation coefficient can be determined. The previous sonar image is generated from the return intensities, which is the base pattern from which the matching process is conducted. The estimated rotation and translation is applied to the previous images through the Euclidean transformation. How closely the two sonar images match is then determined using the correlation coefficient. The correlation coefficient is computed between the previous sonar image (A) and the current sonar image (B) where both A and B are matrices of size m by n.

$$rho = \frac{\sum_m \sum_n \left(A_{mn} - \bar{A}\right)\left(B_{mn} - \bar{B}\right)}{\sqrt{\left(\sum_m \sum_n \left(A_{mn} - \bar{A}\right)^2\right)\left(\sum_m \sum_n \left(B_{mn} - \bar{B}\right)^2\right)}} \tag{16}$$

Once the maximum correlation coefficient is determined, then the best estimate of the rotation and translation is known. From the optimal transformation the velocity and heading rate estimates can be determined. The rotation that was applied directly corresponds to the change in heading of the vehicle and the translation applied

corresponds directly to the amount of forward and lateral distance traveled. The difference in time between the two sonar images is used to convert those distances traveled and the change in heading into an estimate in the vehicles velocity and heading rate during the period between the sonar images. The estimated velocities and heading rates could then be used as inputs into the vehicles steering model to estimate the vehicles motion and position.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. STEERING MODEL

A. INTRODUCTION

Rigid body models are formed in order to analyze, predict, and control motion behavior of autonomous machines that travel over land, air, and undersea. Each type of vehicle model differs in only the terms of the forces applied to produce motion. However, these forces are often controllable and can thus be studied from a prospective of stabilization. This chapter will only deal with the modeling of underwater vehicles. The approach taken with underwater vehicles is that of a moving body in free space without constraint. The forces applied to underwater vehicles include the following: inertial, gravitational, hydrostatic, propulsion, thruster, and hydrodynamic lift and drag forces. (Healey class notes).

B. EQUATIONS OF MOTION IN THE HORIZONTAL PLANE

The following paragraphs describe a simplified development of the steering model used to control the REMUS vehicle. For a more detailed development, see (Healey, 1995). This model was adapted from that of the Acoustic Radio Interactive Exploratory Server (ARIES) AUV (Healey and Marco, 2001) and is based on the following assumptions:

- the vehicle behaves as a rigid body
- the earth's rotation is negligible for acceleration components of the vehicle's center of mass
- the primary forces that act on the vehicle are inertial and gravitational in origin and are derived from hydrostatic, propulsion, thruster, and hydrodynamic lift and drag forces.

Before describing the equations of motion (EOM) that govern the REMUS steering model, a coordinate system for the vehicle and its surrounding area must be defined. The EOM are derived using a Newton-Euler approach that relates the position

and motions in the local plane to those in the global plane. The geometry of the local and global coordinate system can be seen in Figure 13.

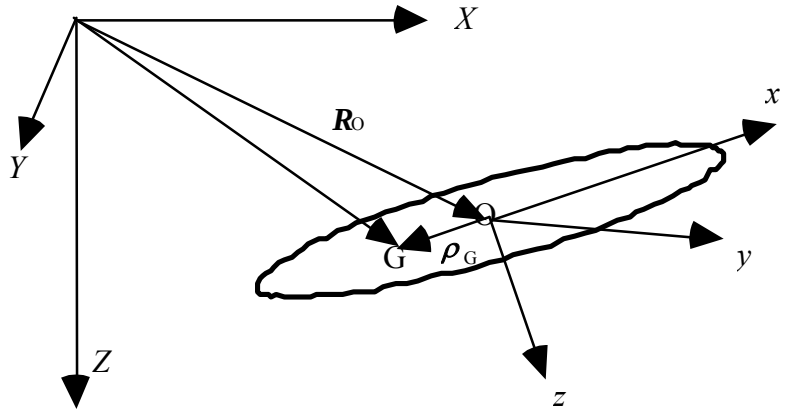

Figure 13. Local and Global Coordinate System (From Marco and Healey, 2001)

In order to convert from a local velocity vector $[u, v, w]$, where u is surge, v is sway, and w is heave, to a global velocity vector $[\dot{X}, \dot{Y}, \dot{Z}]$, a transformation matrix containing 'Euler' angles (ϕ, θ, ψ) must be defined. The transformation matrix (T) is defined as follows:

$$\mathbf{T}(\phi,\theta,\psi) = \begin{bmatrix} \cos\psi\cos\theta, & \sin\psi\cos\theta, & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi, & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi, & \cos\theta\sin\phi \\ \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi, & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi, & \cos\theta\cos\phi \end{bmatrix} \quad (17)$$

Transformation from a global velocity vector to the local velocity vector occurs as follows:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \mathbf{T}(\phi,\theta,\psi) \bullet \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} \quad (18)$$

Transformation from a local velocity vector to a global velocity vector occurs as follows:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \mathbf{T}^{-1}(\phi,\theta,\psi) \bullet \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{19}$$

The global angular velocity vector $[p,q,r]$ can be transformed into the rates of change of the 'Euler' angles as follows:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{20}$$

Healey (1995) derives the equations of motion for a six degree model as:

SURGE EQUATION OF MOTION

$$m\left[\dot{u}_r - v_r r + w_r q - x_G\left(q^2 + r^2\right) + y_G\left(pq - \dot{r}\right) + z_G\left(pr + \dot{q}\right)\right] + \left(W - B\right)\sin\theta = X_f \tag{21}$$

SWAY EQUATION OF MOTION

$$m\left[\dot{v}_r + u_r r - w_r p + x_G\left(pq + \dot{r}\right) - y_G\left(p^2 + r^2\right) + z_G\left(qr - \dot{p}\right)\right] - \left(W - B\right)\cos\theta\sin\phi = Y_f \tag{22}$$

HEAVE EQUATION OF MOTION

$$m\left[\dot{w}_r - u_r q + v_r p + x_G\left(pr - \dot{q}\right) + y_G\left(qr + \dot{p}\right) - z_G\left(p^2 + q^2\right)\right] + \left(W - B\right)\cos\theta\cos\phi = Z_f \tag{23}$$

ROLL EQUATION OF MOTION

$$I_x\dot{p} + \left(I_z - I_y\right)qr + I_{xy}\left(pr - \dot{q}\right) - I_{yz}\left(q^2 - r^2\right) - I_{xz}\left(pq + \dot{r}\right) + m\left[y_G\left(\dot{w} - u_r q + v_r p\right)\right.$$

$$\left. - z_G\left(\dot{v}_r + u_r r - w_r p\right)\right] - \left(y_G W - y_B B\right)\cos\theta\cos\phi + \left(z_G W - z_B B\right)\cos\theta\sin\phi = K_f \tag{24}$$

PITCH EQUATION OF MOTION

$$I_y\dot{q} + \left(I_z - I_z\right)pr - I_{xy}\left(qr + \dot{p}\right) + I_{yz}\left(pq - \dot{r}\right) + I_{xz}\left(p^2 - r^2\right) - m\left[x_G\left(\dot{w} - u_r q + v_r p\right)\right.$$

$$\left. - z_G\left(\dot{u}_r - v_r r + w_r q\right)\right] + \left(x_G W - x_B B\right)\cos\theta\cos\phi + \left(z_G W - z_B B\right)\sin\theta = M_f \tag{25}$$

YAW EQUATION OF MOTION

$$I_z\dot{r}+\left(I_y-I_x\right)pq-I_{xy}\left(p^2-q^2\right)-I_{yz}\left(pr+\dot{q}\right)+I_{xz}\left(qr-\dot{p}\right)+m\left[x_G\left(\dot{v}_r+u_r r-w_r p\right)\right.$$

$$\left.-y_G\left(\dot{u}_r-v_r r+w_r q\right)\right]-\left(x_G W-x_B B\right)\cos\theta\sin\phi-\left(y_G W-y_B B\right)\sin\theta=N_f \qquad (26)$$

Where:

W = weight

B = buoyancy

I = mass moment of inertia terms

u_r, v_r, w_r = component velocities for a body fixed system with respect to the water

p, q, r = component angular velocities for a body fixed system

x_B, y_B, z_B = position difference between geometric center and center of buoyancy

x_G, y_G, z_G = position difference between geometric center and center of gravity

X_f, Y_f, Z_f, K_F, M_f, N_f = sums of all external forces acting in the particular body fixed direction

In addition, he presents a simplified version of these equations of motion. In order to simplify the initial equations of motions the following assumptions were made:

- the center of mass of the vehicle lies below the origin
- x_G and y_G are zero
- the vehicle is symmetric in its inertial properties
- motions in the vertical plane are negligible (i.e., [w_r, p, q, r, Z, ϕ, θ]=0)
- u_r equals the forward speed, U_o

The simplified equations of motion are thus:

$$u_r = U_o \qquad (27)$$

$$m\dot{v}_r = -mU_o r + \Delta Y_f(t) \qquad (28)$$

$$I_{zz}\dot{r} = \Delta N_f(t) \qquad (29)$$

$$\dot{\psi} = r \qquad (30)$$

$$\dot{X} = U_o \cos\psi - v_r \sin\psi + U_{cx} \tag{31}$$

$$\dot{Y} = U_o \sin\psi - v_r \cos\psi + U_{cy} \tag{32}$$

C. HYDRODYNAMIC COEFFICIENTS

Healey proposes that due to symmetry of the vehicle, one can heuristically determine that only a subset of motions would affect the loading in any particular direction (Healey class notes) and uses the following expressions to describe hydrodynamic forces of sway and yaw:

$$\Delta Y_f = f(v_r, dv_r/dt, r, dr/dt, p, dp/dt, t) \tag{33}$$

$$\Delta N_f = f(p, dp/dt, v_r, dv_r/dt, r, dr/dt, t) \tag{34}$$

Sway, yaw, and roll motions are coupled. However, roll motion is often only coupled one way and not considered when evaluating horizontal plane steering. The hydrodynamic forces for sway and yaw are linearized using Taylor series expansion to determine 'hydrodynamic coefficients.' The coefficients are dependent on the shape characteristics of the vehicle and have significant affect on the natural stability of the vehicle. The expression for the transverse (sway) force is:

$$Y_f = Y_{\dot{v}_r}\dot{v}_r + Y_{v_r}v_r + Y_{\dot{r}}\dot{r} + Y_r r \tag{35}$$

and the expression for rotational (yaw) force is:

$$N_f = N_{\dot{v}_r}\dot{v}_r + N_{v_r}v_r + N_{\dot{r}}\dot{r} + N_r r \tag{36}$$

This leads to:

$$Y_{v_r} = \frac{\partial Y_f}{\partial v_r};\ \ \mathrm{Y}_r = \frac{\partial N_f}{\partial r};\ \ \mathrm{Y}_{\dot{v}_r} = \frac{\partial Y_f}{\partial \dot{v}_r};\ \ \mathrm{Y}_{\dot{r}} = \frac{\partial Y_f}{\partial \dot{r}}; \tag{37}$$

and

$$N_{v_r} = \frac{\partial N_f}{\partial v_r};\ \ \mathrm{N}_r = \frac{\partial N_f}{\partial r};\ \ \mathrm{N}_{\dot{v}_r} = \frac{\partial N_f}{\partial \dot{v}_r};\ \ \mathrm{N}_{\dot{r}} = \frac{\partial N_f}{\partial \dot{r}}; \tag{38}$$

Where:

$Y_{\dot{v}_r}$ = coefficient for added mass in sway

$Y_{\dot{r}}$ = coefficient for added mass in yaw

Y_{v_r} = coefficient of sway force induced by side slip

Y_r = coefficient of sway force induced by yaw

$N_{\dot{v}_r}$ = coefficient for added mass moment of inertia in sway

$N_{\dot{r}}$ = coefficient for added mass moment of inertia in yaw

N_{v_r} = coefficient of sway moment from side slip

N_r = coefficient of sway moment from yaw

The hydrodynamic coefficients for steering for the REMUS vehicle were adapted from thesis work performed by Massachusetts Institute of Technology (Prestero, 2001) establishing estimates of all vehicle coefficients. Upon re-calculation, Fodrea (2002) adjusted the hydrodynamic coefficients to account for variation in experimental data. Table 2 lists the REMUS hydrodynamic coefficients for the steering model used during this experiment.

$Y_{\dot{v}_r}$	-3.55e01 kg
$Y_{\dot{r}}$	1.93 kg m/rad
Y_{v_r}	-6.66e01 kg/s (Same as Zw)
Y_r	2.2 kg m/s (Same as Zq)
$N_{\dot{v}_r}$	1.93 kg m
$N_{\dot{r}}$	-4.88 kg m^2/rad
N_{v_r}	-4.47 kg m/s
N_r	-6.87 kg m^2/s (Same as Mq)
N_δ	-3.46e01/3.5 kg m/s^2
Y_δ	5.06e01/3.5 kg m/s^2

Table 3. REMUS Hydrodynamic Coefficients for Steering (From Fodrea, 2002)

The dynamics of the vehicles are defined as:

$$m\dot{v}_r = -mr + Y_{\dot{v}_r}\dot{v}_r + Y_{v_r}v_r + Y_{\dot{r}}\dot{r} + Y_r r + Y_\delta \delta_r(t) \tag{39}$$

$$I_{zz}\dot{r} = N_{\dot{v}_r}\dot{v}_r + N_{v_r}v_r + N_{\dot{r}}\dot{r} + N_r r + N_\delta \delta_r(t) \tag{40}$$

$$\dot{\psi} = r \tag{41}$$

D. VEHICLE KINEMATICS

The kinematics of the vehicle is described by Equations (39) and (40). U_{cx} and U_{cy} are the current velocities in the associated direction. The kinematic equations, along with the heading rate, compose the steering dynamics of REMUS and can be expressed as follows:

$$\begin{bmatrix} m-Y_{\dot{v}_r} & -Y_{\dot{r}} & 0 \\ -N_{\dot{v}_r} & I_{zz}-N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v}_r \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_{v_r} & Y_r - mU_0 & 0 \\ N_{v_r} & N_r & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} Y_\delta \\ N_\delta \\ 0 \end{bmatrix} \delta_r(t) \tag{42}$$

where $\delta_r(t)$ represents the control input for both rudders.

E. VEHICLE DYNAMICS

The final assumption made for vehicle dynamics (Johnson, 2001) is that the cross coupling terms in the mass matrix is zero. Thus, the final vehicle dynamics are defined as:

$$\begin{bmatrix} m-Y_{\dot{v}_r} & 0 & 0 \\ 0 & I_{zz}-N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v}_r \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_{v_r} & Y_r - mU_0 & 0 \\ N_{v_r} & N_r & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} Y_\delta \\ N_\delta \\ 0 \end{bmatrix} \delta_r(t) \tag{43}$$

F. APPLICATION

The ultimate goal of using the proposed method of estimating vehicle velocities and heading rate is to accurately navigate in the undersea environment. Once a real-time velocity estimate can be provided and employed in the AUVs navigation and position estimation systems, these inputs could be used vice the ADCP measurements. The velocities and heading rate estimates from the FLS could be used to update the vehicle motion model and state information (Figure 14). The state information of the vehicle contains the vehicle kinematics. The kinematic equations, along with the heading rate, compose the steering dynamics of REMUS AUV.

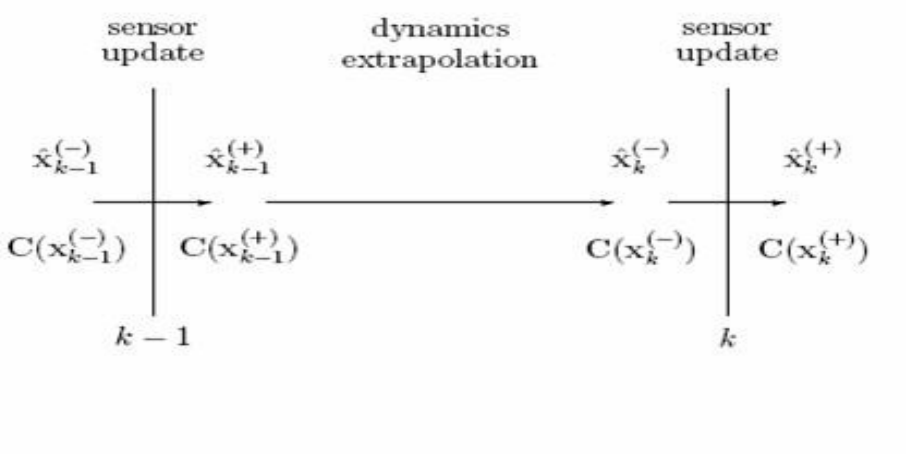

Figure 14. Using Sensor Updates to Update State Information (From Smith, Self, and Cheeseman, 1990)

THIS PAGE INTENTIONALLY LEFT BLANK

V. SIMULATION AND RESULTS

A. INTRODUCTION

The proposed method of estimating velocities from sequential sonar images is applied to previously recorded data. This method assumes that all features within an image are stationary, and therefore the relative local intensities remain constant. The intensities of the returns within the sonar image are dependent upon range, bearing, and time I (r,θ,t). The AUVs velocity estimate is a determination of the motion between images. The change in intensity over range and bearing with respect to time, dr/dt and dθ/dt, is used to perform motion analysis. The displacements of those features, as detected within the images through their intensity, can be converted to an estimation of the velocity of the AUV in both the forward and lateral directions. In essence, each feature, as identified through intensity, within the first image is matched to the same intensity within the second image.

The AUV velocity estimate is applied using Euclidean transformations. This transformation method preserves length and angle, so the shape of an image is not distorted: straight lines transform to straight lines, planes transform to planes and circles transform to circles, for example. Only the position and orientation of the object changes, which is why they are also called rigid body transformations. Euclidean transformations applied in this thesis can be decomposed into two operations, rotation and translation. The process of rotation and translation are applied to the images within the image space. Within an image there is a defined region which is the sonar's field of view; these are the only applicable pixels. Therefore a modification to the transformation must be applied, since non-applicable pixels must be excluded from the correlation coefficient. Since some pixels will be outside the vehicles field of view once the velocity estimates are applied, those pixels must not be used in the calculation of the correlation coefficient.

The correlation coefficient is then calculated, comparing the two sequential sonar images with the transformation applying a velocity estimate. A search is then used to

maximize the correlation coefficient. Once the maximum correlation coefficient is determined then the translation and rotation are converted back to velocity estimates for the REMUS AUV.

B. MODEL PROCESS

The velocity estimation process was run on sonar image and data collected from the REMUS AUV. The FLS images that were analyzed were collected on 25 January 2006 in Monterey Bay, California. The specific area selected was on average 17 meters deep, ranging from 15 to 21 meters, with a sandy bottom. The data was chosen in part because the three to four meter surf represented a challenging navigation environment for the vehicle. The mission was a minefield survey; therefore several mine-like objects were deployed to simulate a mine field. Divers visually observed small ripples and pock marks within the relatively flat sandy bottom, and occasionally noted sparse kelp growing from the bottom. The REMUS AUV was deployed and executed a preprogrammed mission (Figure 15).

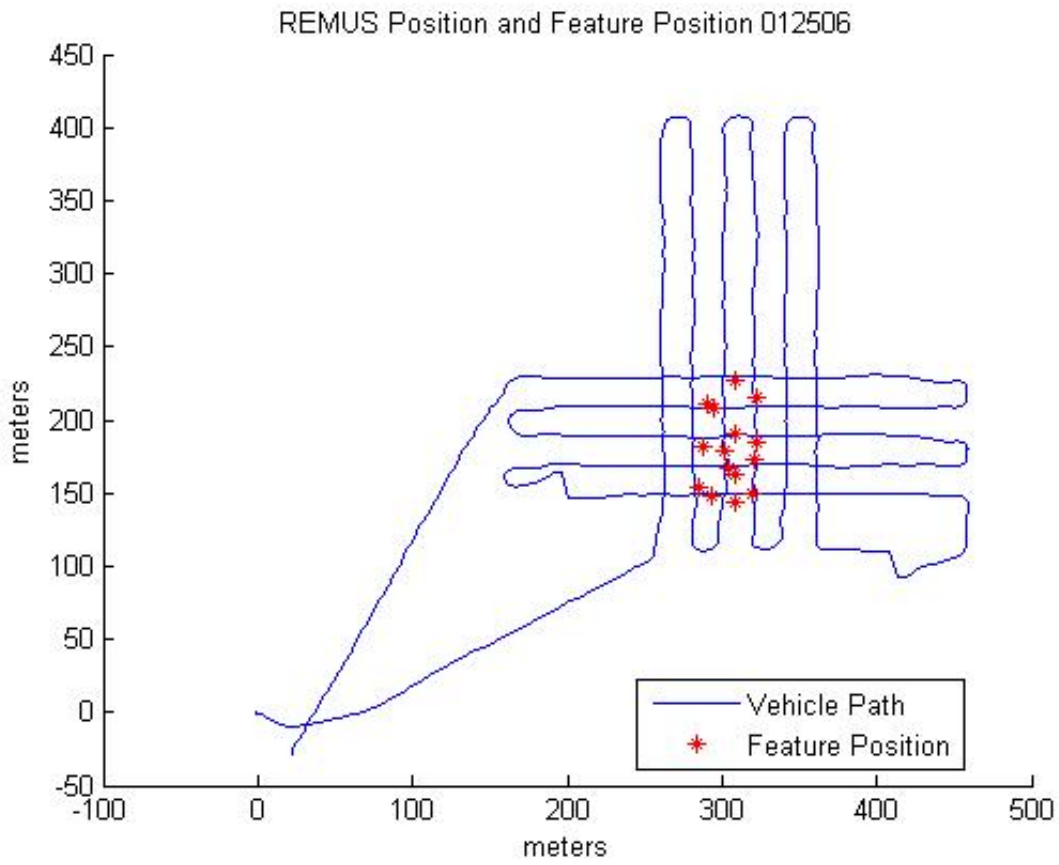

Figure 15. REMUS AUV Position and Feature Positions 012506

As can be seen in Figure 15 the mission was a mine field survey using a technique commonly referred to as "mowing the lawn". The vehicle transited the minefield and then conducts two ninety degree turns to perform another parallel transit of the minefield. These transits lanes are specifically designed based on the sensor width, to ensure proper coverage of the minefield. After deployment the vehicle dove to the preprogrammed depth and conducted the mission at an average altitude of 3.3 meters. The data collected by the REMUS AUV was saved into a Mathworks Matlab© structure "state_lbl_adcp_pings012506_01.mat". (Table 4). In the example structure it can be seen that some of the fields are "-999". This is the entry that is used when there is no information available during that time. Not all of the fields update at the same frequency, and the "-999" is inserted when there is no new information in that field. All of the sonar images collected during the mission were saved in a folder called "Allpings" as .son files.

The Blueview ProVeiwer software saves the entire mission as a compressed record in a single file. Blueview Technologies software ProViewer was used to convert the proprietary .son files to xml .raw files. By exporting and converting the .son to .raw files the individual images can be easily accessed for comparison.

lblLatitude:	-999
lblLongitude:	-999
adcpLatitude:	36.7169
adcpLongitude:	-121.82
forwardVelocity:	-0.1359
starboardVelocity:	0.0689
verticalVelocity:	-0.062
altitude:	19.1839
latitude:	-999
longitude:	-999
depth:	-999
compassHeading:	-999
headingRate:	-999
estimatedVelocity:	-999
pitch:	-999
pitchRate:	-999
roll:	-999
rollRate:	-999
flsFileNumber:	-999
time:	3.16E+04

Table 4. Example Structure Containing Data from REMUS Mission 012506

1. Image Preprocessing

Two previously recorded sequential FLS images are opened for comparison to determine the lateral and forward velocities and heading rate of the REMUS AUV between the images. When the image matching is conducted by maximizing the correlation coefficient the vehicle velocity in the forward direction is approximated to be zero. As can be seen in Figure 16, the value of the correlation coefficient increases as the two images are compared with less forward velocity applied through the Euclidean

transformation. The reduction in forward velocity aligns the high intensity pixels that are created from the near-field effects, thus resulting in forward velocity estimates that are much less than the actual velocity. Another artifact of the near field intensities can be easily seen in Figure 17. As the correlation coefficient is calculated at heading rates where the magnitude is greater than approximately 10 degrees per second it can be observed the correlation coefficient is increasing. This artifact is generated since at the high magnitude heading rates the near-field intensity of one stave is now being aligned with the near-field intensity of the opposite stave in the next image.

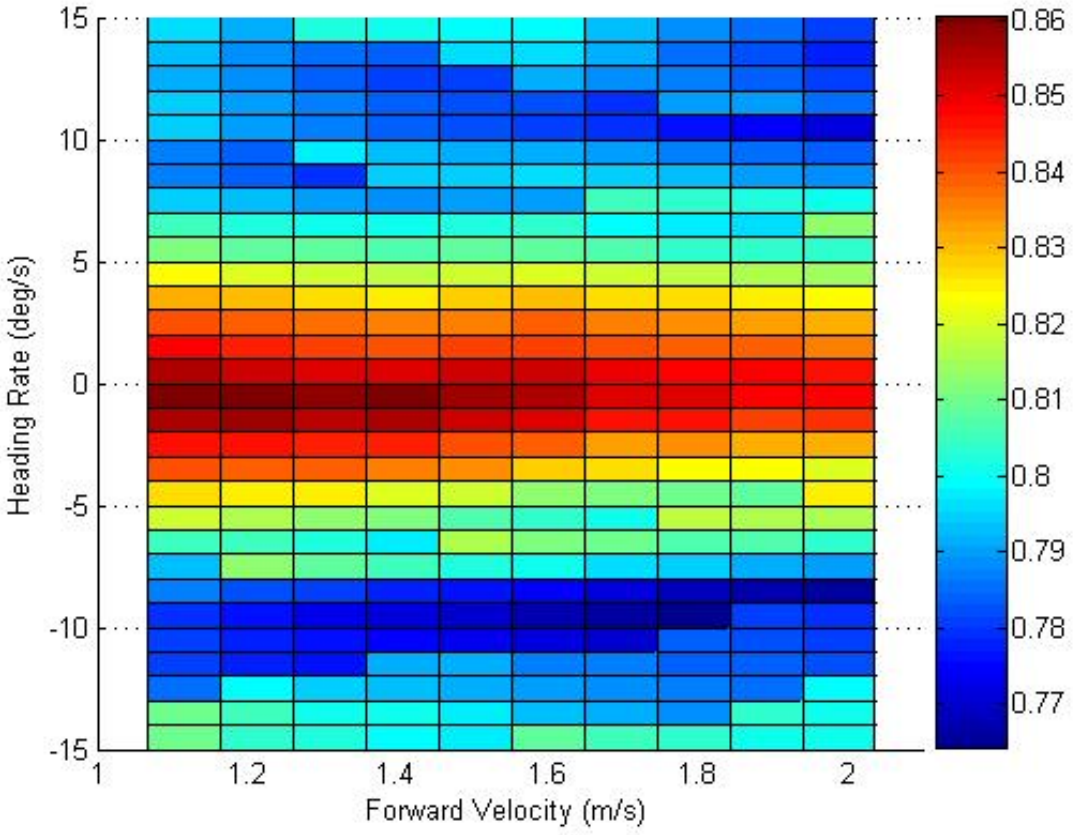

Figure 16. Correlation coefficient versus Forward Velocity and Heading Rate from two sequential FLS images REMUS AUV 012506

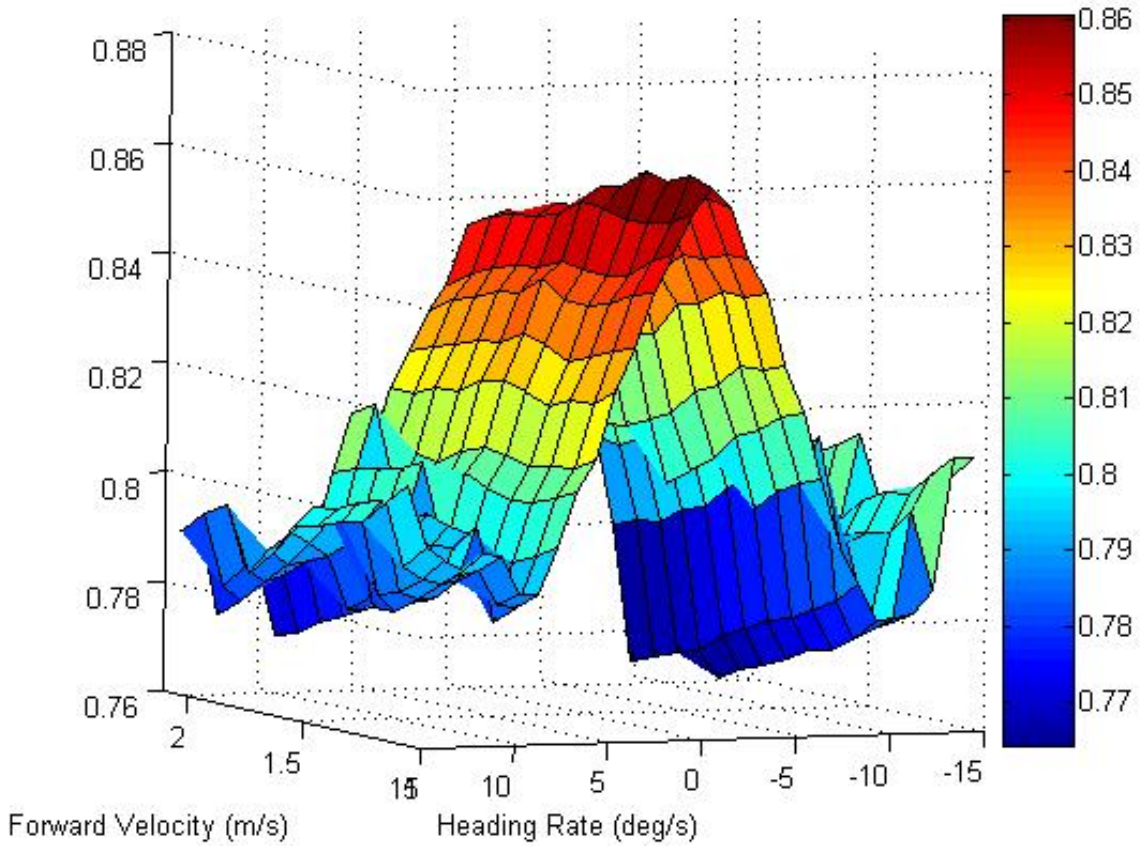

Figure 17. Correlation coefficient versus Forward Velocity and Heading Rate from two sequential FLS images REMUS AUV 012506

To remove the near-field effects from the two sonar images, preprocessing is conducted. Average pixel intensity is determined from the far-field portion of the sonar's field of view. This average pixel intensity is then used to replace the intensity returns of the near field effects. A constant intensity is not utilized since the relative intensity between images is variable.

Another form of preprocessing was considered for the sonar images. The images are currently in Cartesian coordinate system. Converting the sonar images to the Polar coordinate system was considered, however there would have been distortion within the image, especially in the near field region. If the conversion had been applied, a weighting system would have been required to compensate for the distortion. Due to this

distortion, and the additional manipulation that would have been required, it was decided to keep the images in the Cartesian coordinate system.

2. Transformation

Using kinematic estimates, bounded by physical characteristics of the vehicle, the previous sonar image is adjusted using Euclidean translations in the forward (U) and lateral (V) directions. The velocity estimates in the forward direction were bounded from 1.0 to 2.0 meters per second, which is based on physical limitations applied to the vehicle during these experiments and to reduce the processing time of the model. There is a minimum error that can be expected from estimating the forward velocities from the sonar images. The error is dependent upon the resolution of the sonar system employed and the setting used. The FLS Blazed Array used during these tests was set for a range of 90 meters. The images produced based on the resolution settings were matrices of 464 pixel rows and 334 pixel columns. Therefore the range resolution of the FLS in the lowest resolution setting is 0.1940 meters per pixel. Therefore the forward velocity estimates were in increments of 0.2 meters per second, based on these sonar resolutions setting which used for the experiment. The velocity estimate in the forward and lateral direction is converted to a number of pixels. The previous sonar image is adjusted by that number of pixels to simulate motion in the forward and lateral direction between the two sonar images. The adjustment is performed by the number of pixels counted upward from the images origin and the outer edges of the field of view. Then a line is drawn connecting the points and the inapplicable pixels are replaced with zeros.

As can be seen in upper left image in Figure 18 the line is drawn connecting the translated origin and the translated left edge of the field of view. Then in the upper right image the non-applicable pixel intensities are replaced with zeros. The lower left images shows the line drawn connecting the translated origin and the translated right edge of the field of view. The bottom right image shows the fully translated sonar image after the remaining non-applicable pixel intensities are replaced with zeros. Note that the near-field effects have not been removed so the process can be more easily observed.

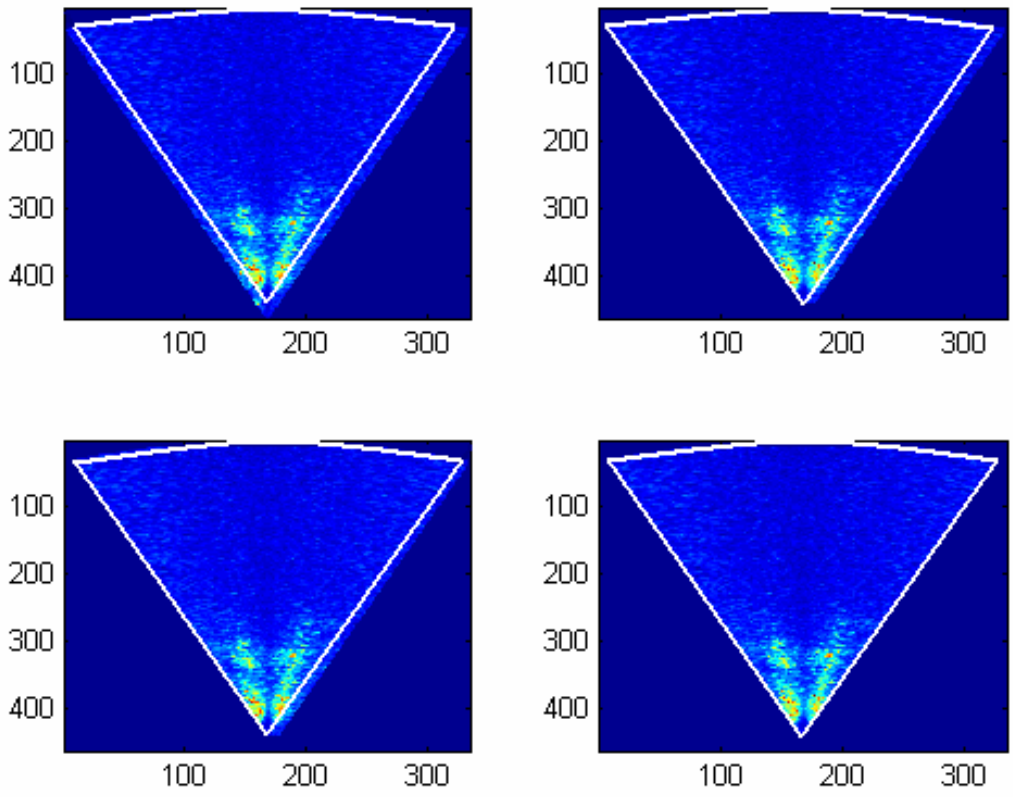

Figure 18. Example of Translation Being Applied to a single FLS Image. Upper left image shows the initial sonar image with the required FOV outlined. Lower right shows the sonar images with the translation applied. Note that the near-field effects have not been removed so the process can be more easily observed.

The heading rate estimate is converted to a number of degrees. The sonar field of view is known. Applying a heading rate will result in pixels which are outside the field of view. The size of the field of view within the two images is used to remove the pixels which would be outside the modified sonar field of view. A point at the maximum width of the field of view is determined based on the assumed heading rate. This point is connected with a line to the image origin. Pixels which would be outside the AUV's field of view are removed. The previous image is then rotated, using Euclidean rotation, that number of degrees to simulate motion of the vehicle between the two sonar images. These steps can be seen in Figures 19 and 20, where the top left image is the translated image and the top right is the image without translation applied. The middle two images

show the lines being drawn connecting the image origin with the appropriate point in the field of view. The bottom two images have the non-applicable pixels removed. Again note that the near-field effects have not been removed so the process can be more easily observed. The two figures show identical magnitude heading rate estimates; however one is clockwise while the other is counter-clockwise.

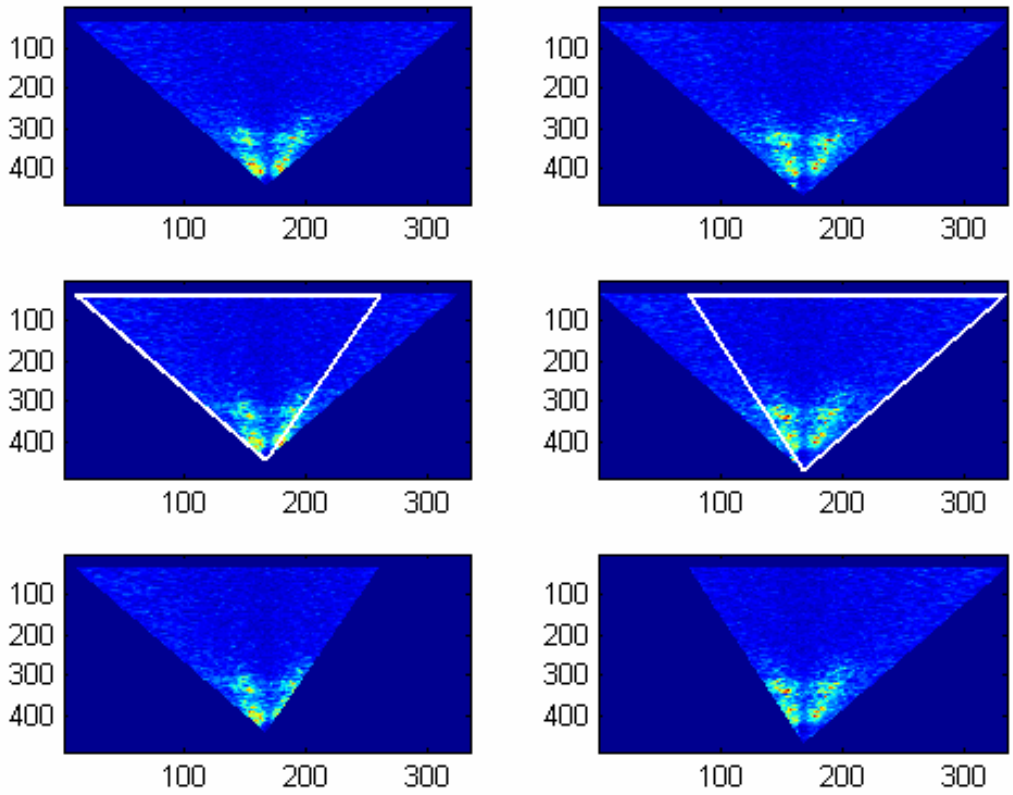

Figure 19. Removing Overlapping Pixels in Preparation for Counter-Clockwise Rotation Being Applied to two FLS Images. The upper left sonar image is the previous image with translation applied, while the upper right is the current sonar image. The white outlines the two FOV to be compared. The bottom images show the FOV with overlapping pixels removed. Note that the near-field effects have not been removed so the process can be more easily observed.

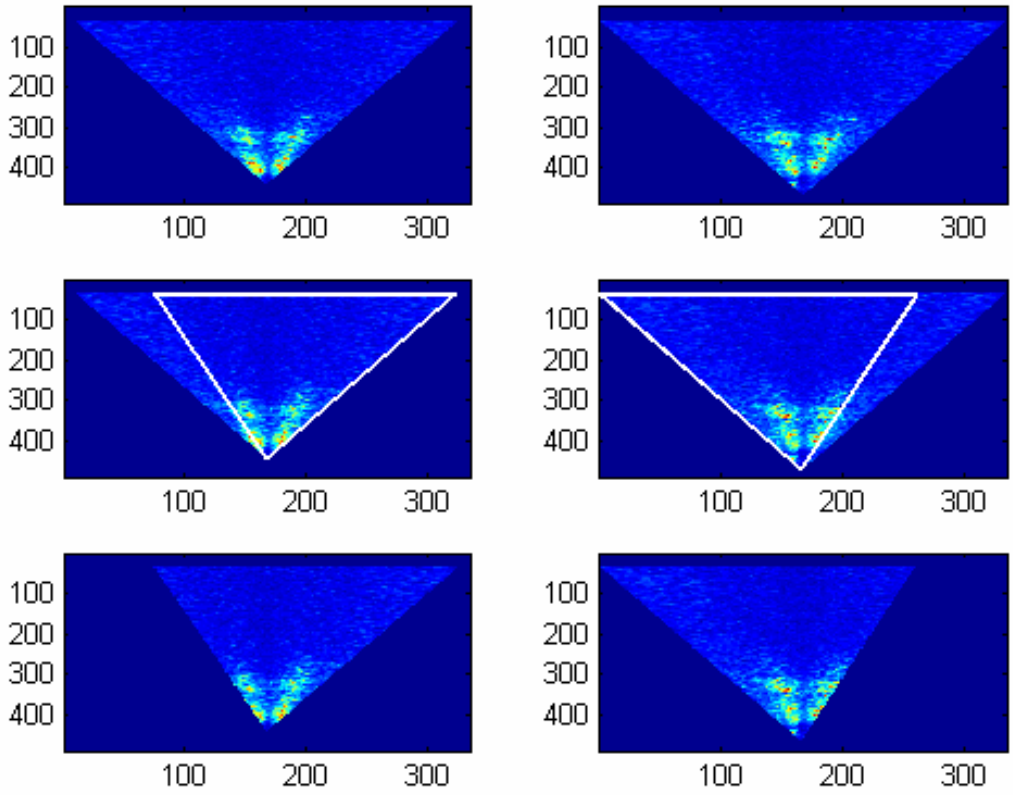

Figure 20. Removing Overlapping Pixels in Preparation for Clockwise Rotation Being Applied to two FLS Images. The upper left sonar image is the previous image with translation applied, while the upper right is the current sonar image. The white outlines the two FOV to be compared. The bottom images show the FOV with overlapping pixels removed. Note that the near-field effects have not been removed so the process can be more easily observed.

Since the rotation is Euclidean there is no image distortion; however the image size increases due to this process. After applying the Euclidean rotation, the image which is rotated consists of a matrix which is larger than the image which is not rotated. The field of view is also no longer centered correctly due to the rotation. This can be seen in Figure 21 where the top left right image need to be rotated and must be compared to the top left image. The next row down shows the rotated image and the image it must be compared to; note the pixel size of the two images. To properly calculate the correlation coefficient from the images, the image is corrected in two steps. First the image origins are adjusted to be at the identical pixel locations, which can be seen in the two images on

the third row. This is done by adding rows and/or columns of zeros as necessary to the appropriate image. The image origins must be in the exact same location to allow for pixel-to-pixel comparisons to occur, which is required for the matching technique. To calculate the correlation coefficient the images must be of identical pixel size. The next step adds rows and/or columns of zeros to the images as required, ensuring the correlation coefficient can be determined. The results can be seen in the bottom two images. Again note that the near-field effects have not been removed so the process can be more easily observed. Now the correlation coefficient can be calculated.

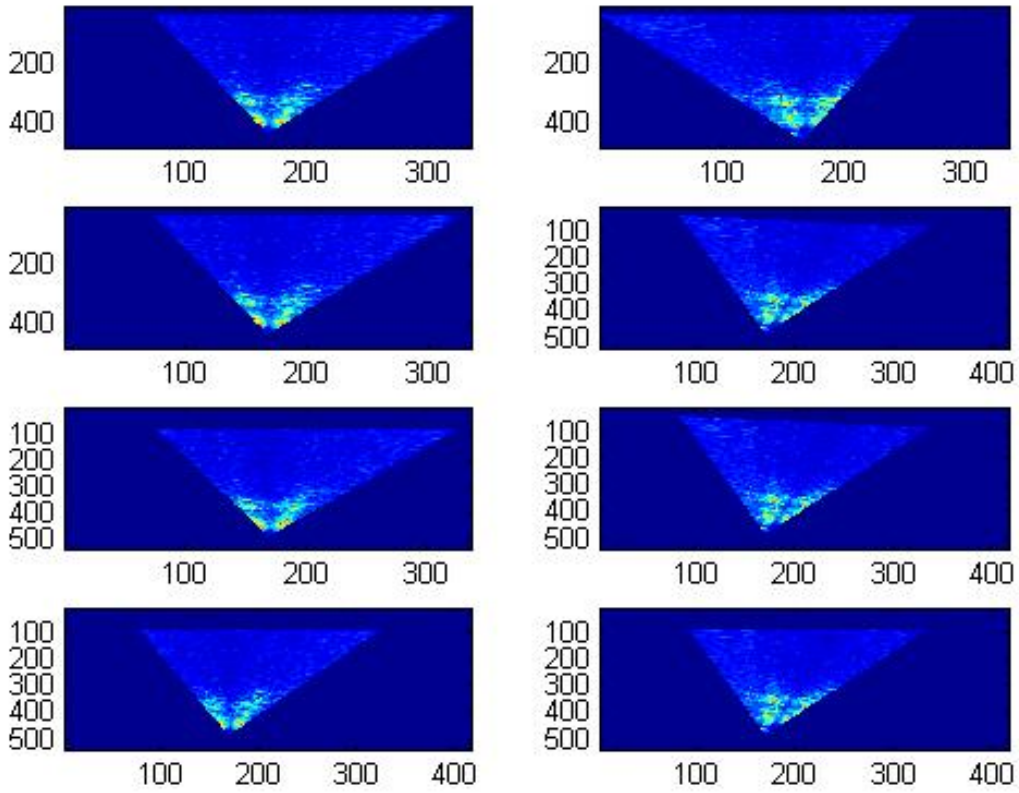

Figure 21. Image Rotation and Removing Overlap to Ensure Proper Calculation of the Correlation Coefficient. The upper two sonar images are to be compared. The second row shows the right sonar image rotated. The third row shows the addition of rows and columns of zeros to match the two images origins. The final row shows the addition of rows and columns of zeros to create two identically sized matrices. Note that the near-field effects have not been removed so the process can be more easily observed.

3. Correlation coefficient

The correlation coefficient is calculated between the modified previous sonar image and the current sonar image. The search process is conducted by first translating the image and then rotating that image incrementally to the right and left. Then the original image is translated an increment larger and the rotations are applied incrementally to the right and left, and so on until the search is completed. At each step the correlation coefficient is calculated, and the maximum is tracked throughout the process. The exhaustive search in will find the combination of translation and rotation which maximizes the correlation coefficient.

4. Motion Analysis

The total translation and rotation modifications that are applied to the sonar image which maximize the correlation coefficient are then converted to an estimate of the velocities in the forward and lateral directions and heading rate. A forward distance can be estimated through a conversion from the number of pixel rows the previous sonar image was modified. Using the time between the sonar images a forward velocity estimate can be determined. The same process is applied to estimate the lateral velocity. The total rotation in degrees which is applied to the previous sonar image and maximizes the correlation coefficient is the turn in degrees conducted between the two sonar images. The time between the two sonar images is used to convert the turn conducted into and estimate of the heading rate.

C. RESULTS

The following sections will present the results of the initial REMUS AUV mission. Additional experimentation was conducted based on those initial results and those results will also be presented.

1. REMUS AUV Mission 012506

During the entire REMUS AUV mission 2420 sonar image comparisons were conducted. The average correlation coefficient was 0.8923 with a maximum value of 0.903 and a minimum value of 0.7698.

The average forward velocity measured by the ADCP DVL was 1.5 meters per second for the duration of the entire mission. The average error through the entire mission between the ADCP measured and imagery-based estimated forward velocities was 0.0392 meters per second, (Appendix A) which results in approximately 2 percent error in the forward velocity estimate compared to the ADCP DVL measurements (Figures 22-33). As previously discussed the image pixel size was 464 by 334, therefore range resolution of the FLS in the lowest resolution setting is 0.1940 meters per pixel. The average magnitude of the error for the entire mission is 0.2291 meters per second, which is approximately the resolution of the sonar.

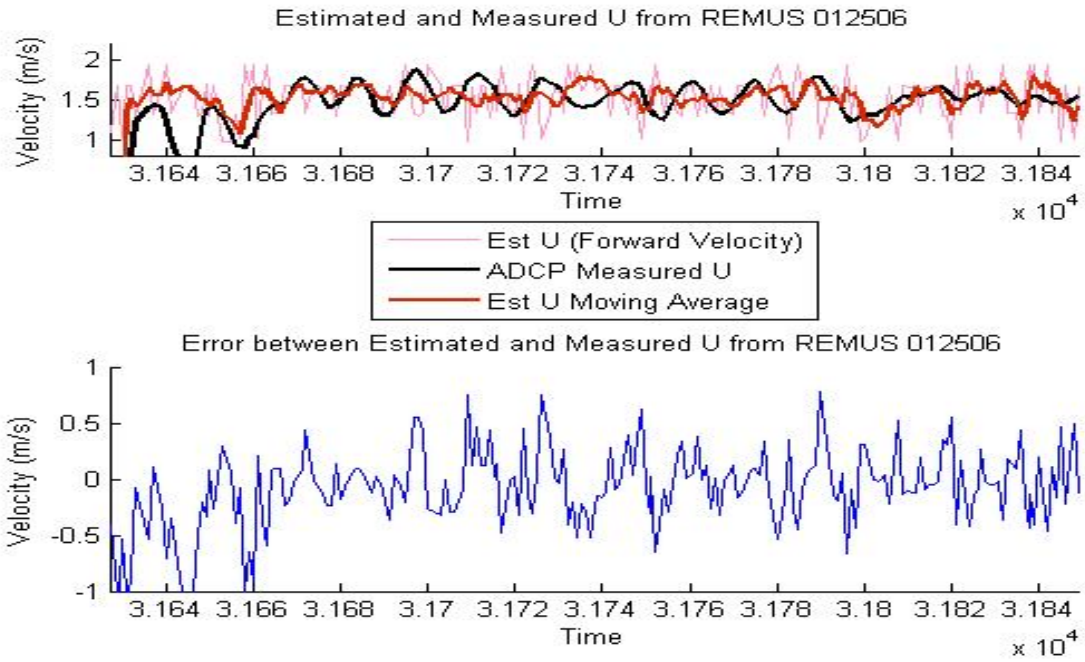

Figure 22. Estimated and Measured U for REMUS 012506 from 31627 to 31849

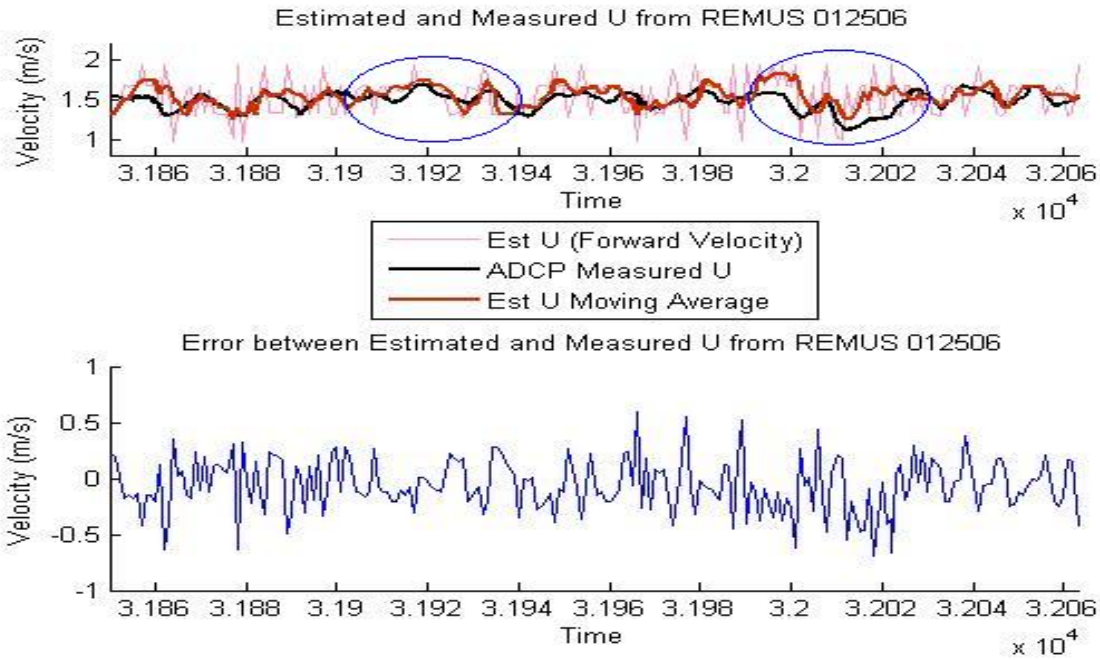

Figure 23. Estimated and Measured U for REMUS 012506 from 31850 to 32063. Note that the first circle identifies when the AUV was in the minefield, the second is identifies when the AUV conducted two ninety degree turns.

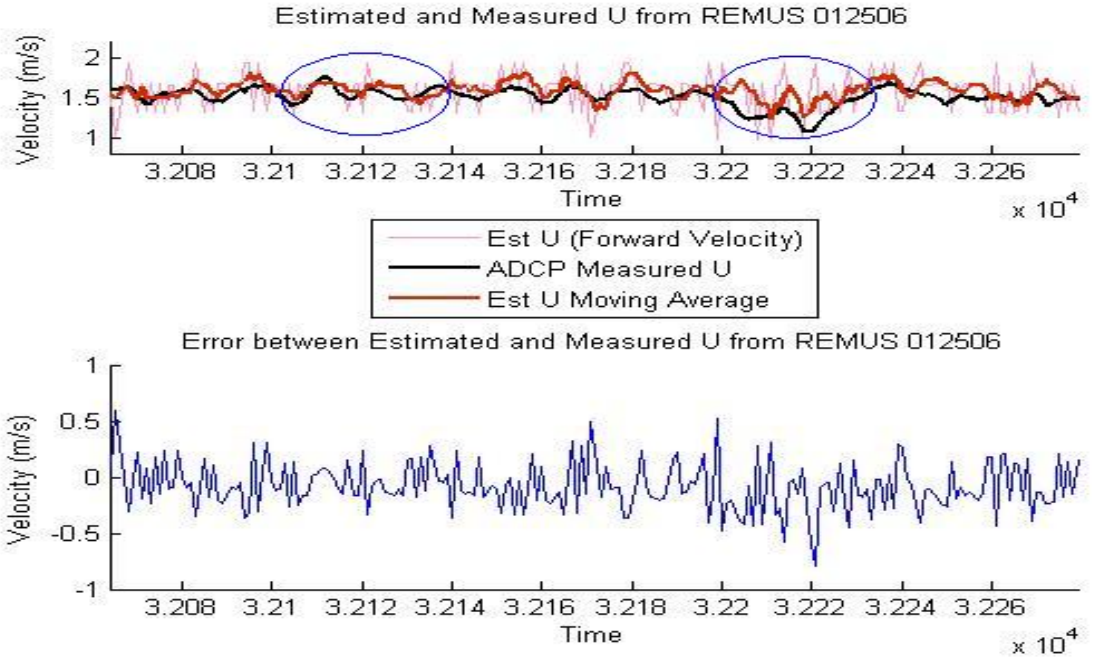

Figure 24. Estimated and Measured U for REMUS 012506 from 32064 to 32279. Note that the first circle identifies when the AUV was in the minefield, the second is identifies when the AUV conducted two ninety degree turns.

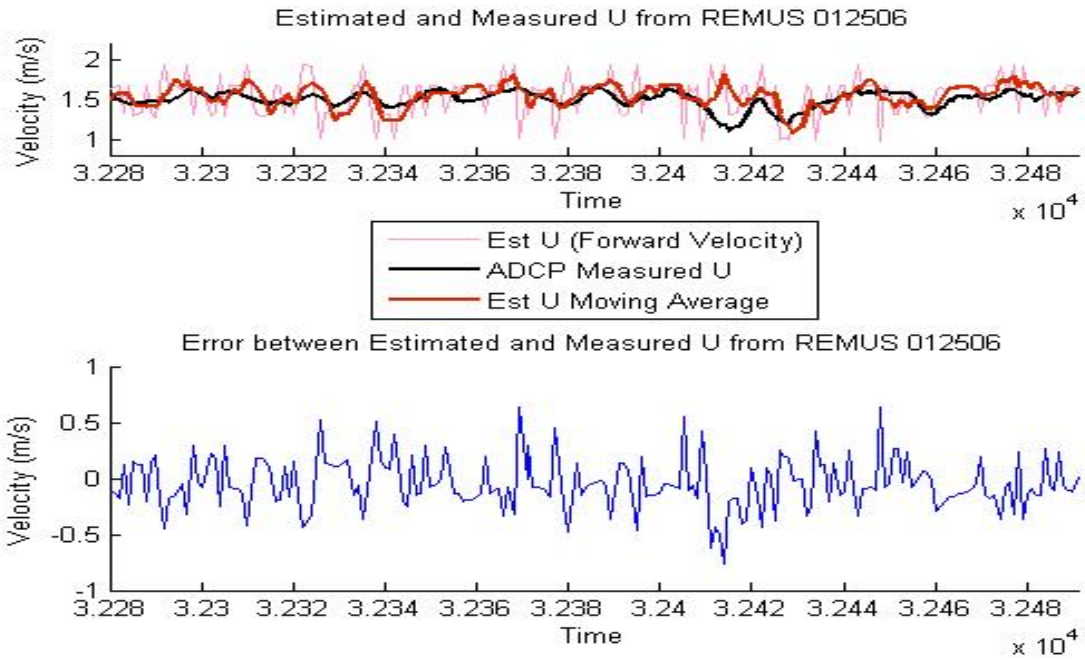

Figure 25. Estimated and Measured U for REMUS 012506 from 32280 to 32491

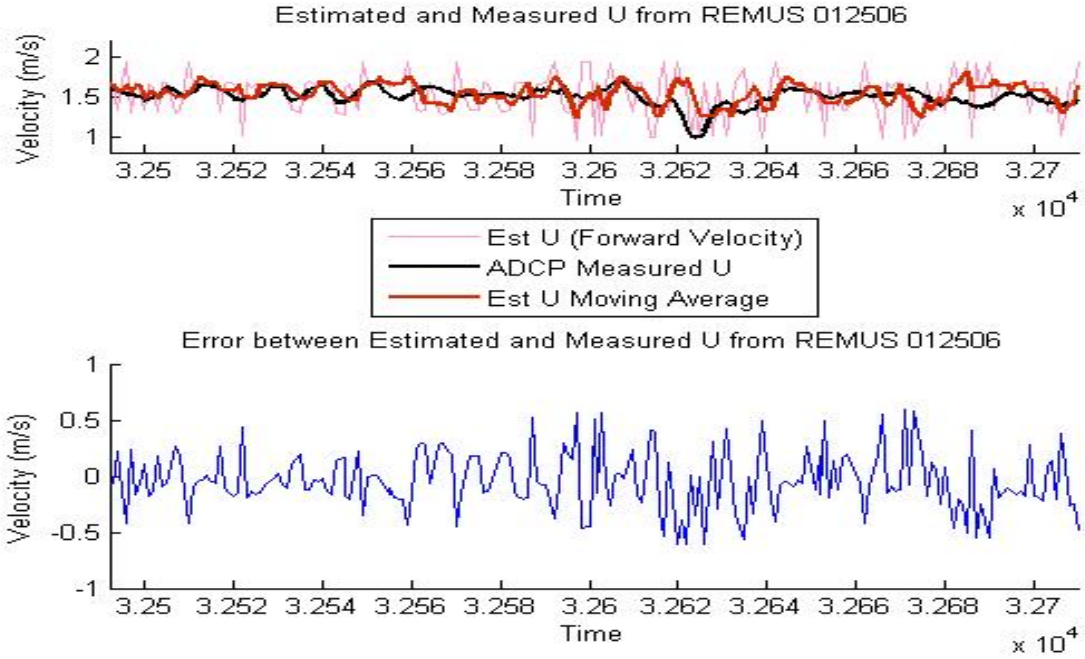

Figure 26. Estimated and Measured U for REMUS 012506 from 32492 to 32710

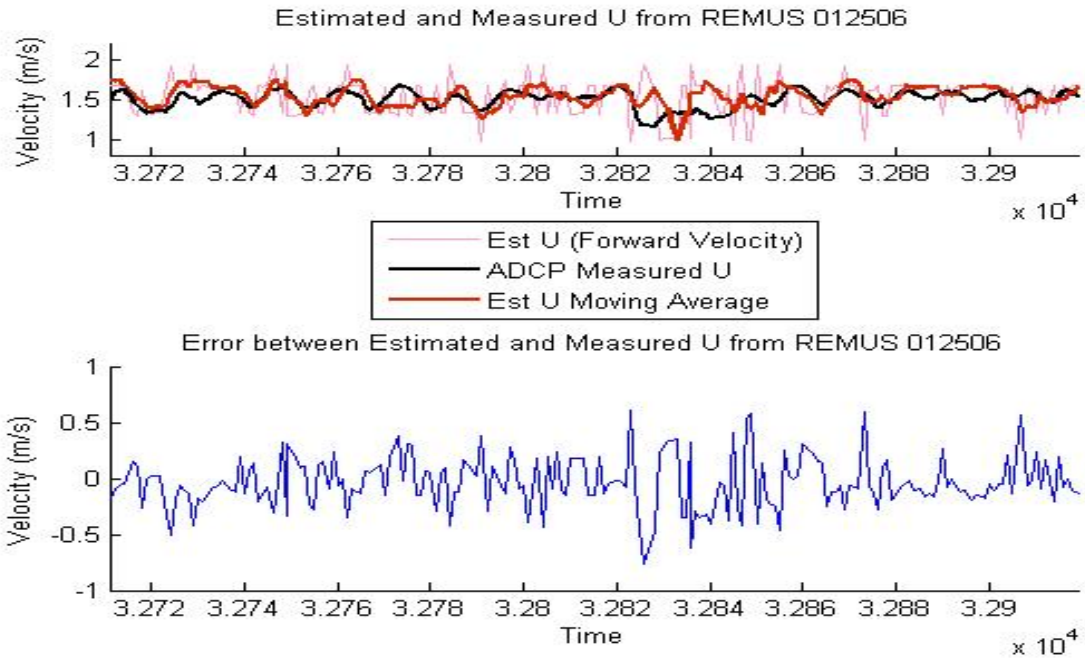

Figure 27. Estimated and Measured U for REMUS 012506 from 32711 to 32919

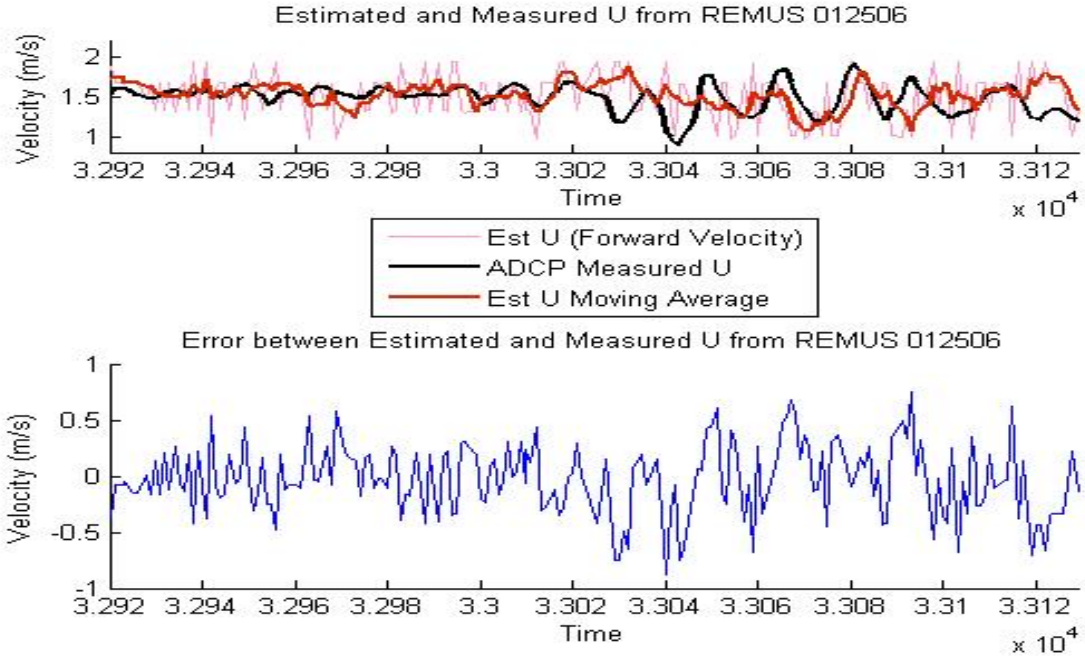

Figure 28. Estimated and Measured U for REMUS 012506 from 32920 to 33129

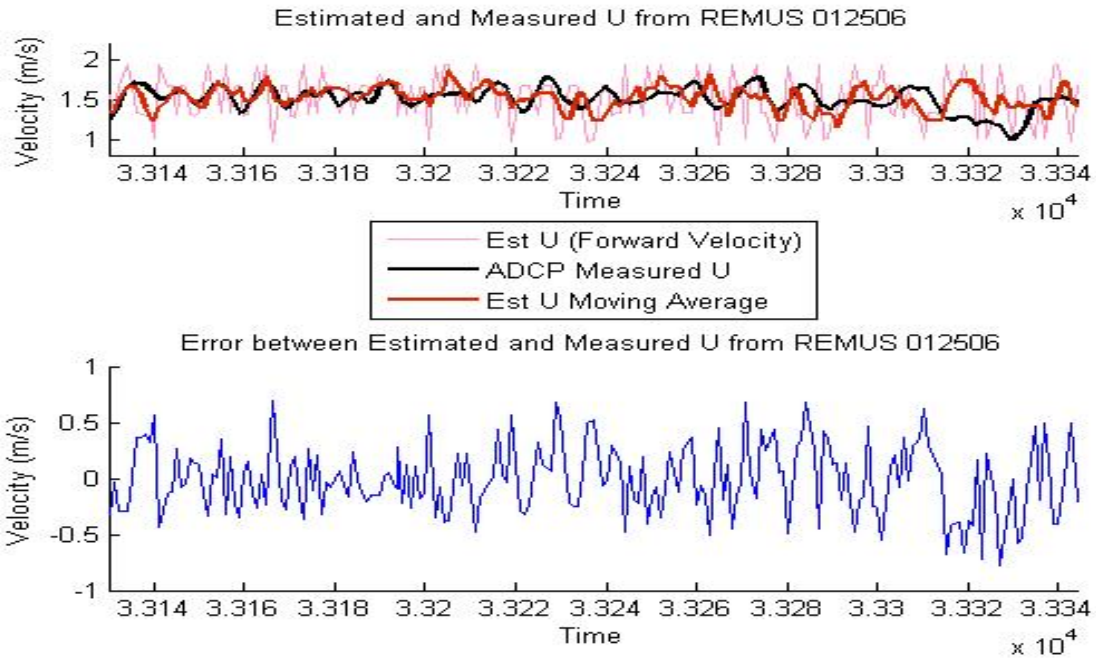

Figure 29. Estimated and Measured U for REMUS 012506 from 33130 to 33344

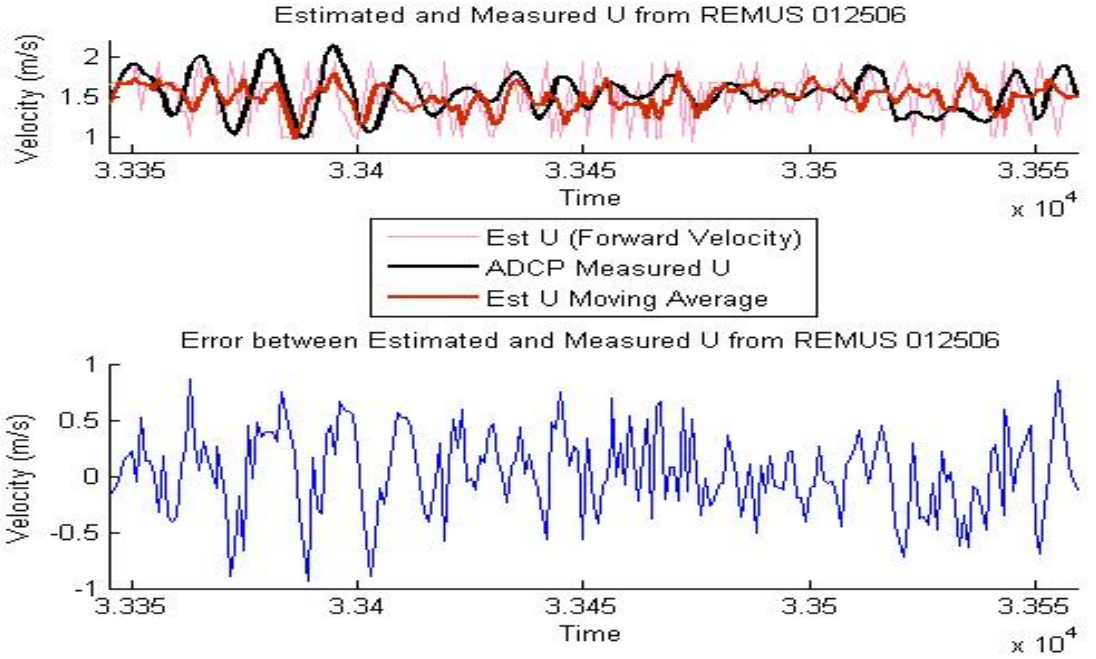

Figure 30. Estimated and Measured U for REMUS 012506 from 33345 to 33559

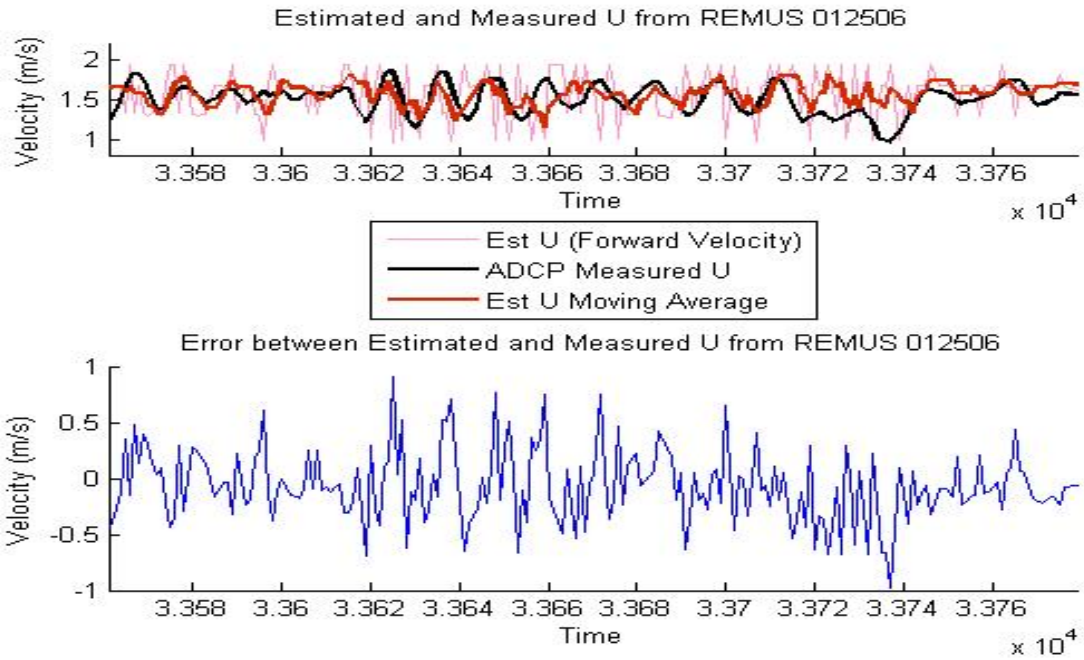

Figure 31. Estimated and Measured U for REMUS 012506 from 33561 to 33779

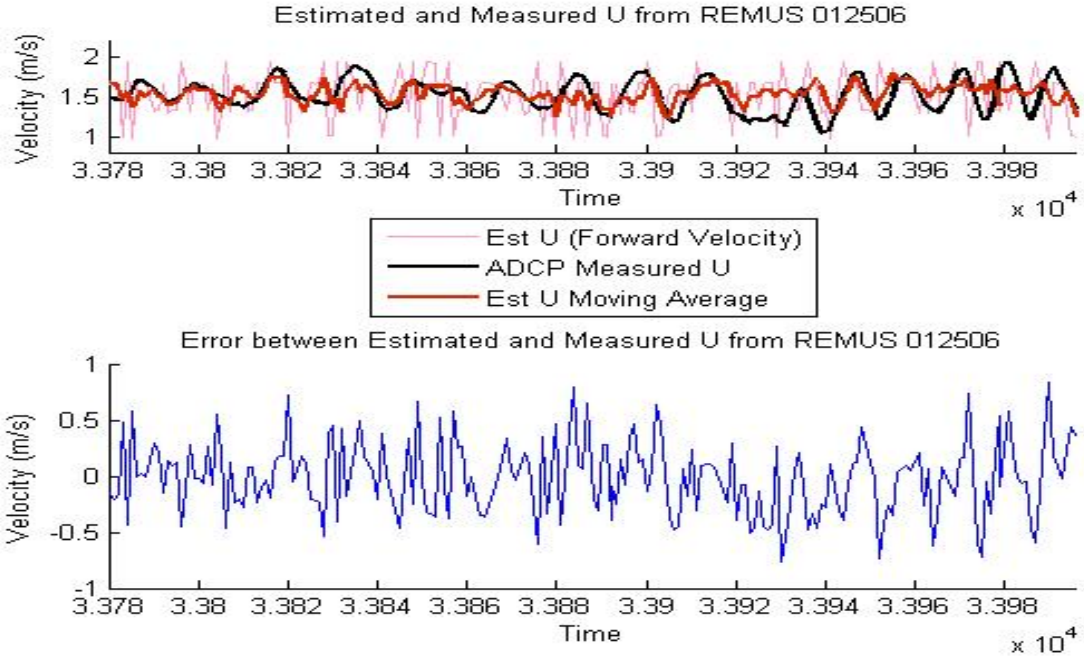

Figure 32. Estimated and Measured U for REMUS 012506 from 33780 to 33996

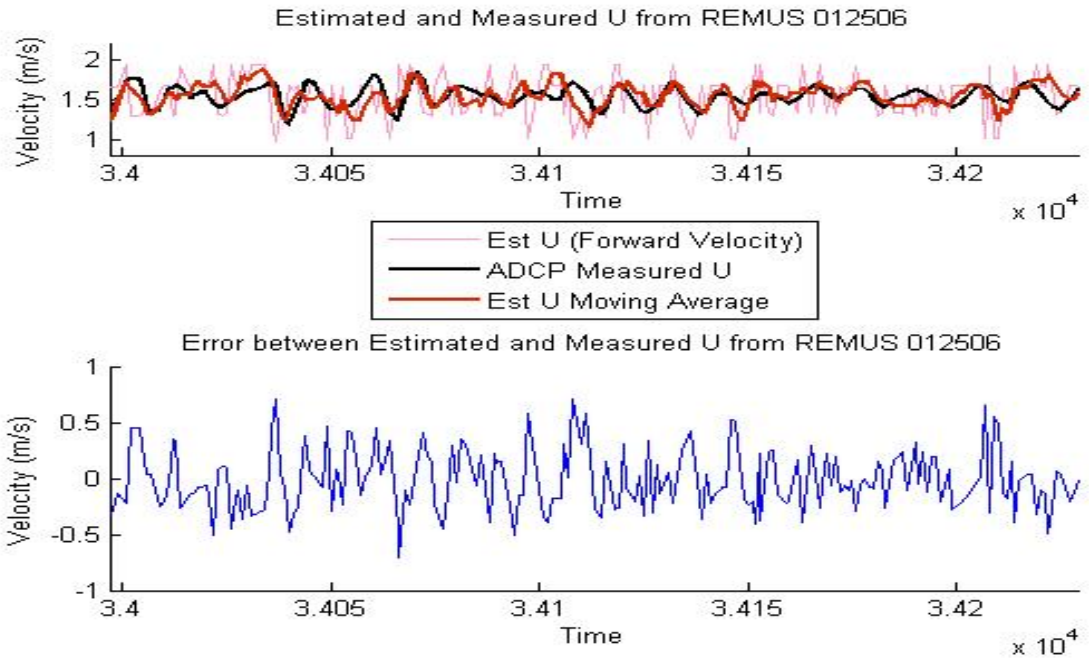

Figure 33. Estimated and Measured U for REMUS 012506 from 33997 to 34229

As can be seen by the ADCP DVL measured forward velocity, the AUV velocity has a sinusoidal component around the average speed, probably due to wave action on the vehicle. It can also be seen that the forward velocity is reduced through the turns by the increased drag on the vehicle. Examining the sonar image based velocity estimates these velocity changes through the turns can also be observed. The sinusoidal component around the average speed can also be seen when an average trend line is plotted. As was discussed earlier, due to the characteristics of the sonar imagery, noise is also easily seen in the data. However a noticeable characteristic of the data is that there is significantly less noise associated with the velocity estimates when the vehicle transited the minefield. This is due to the stronger intensities associated with the mine-like features within the sonar images. This can be seen, for example, in Figures 23 and 24. At time 31910 the AUV entered the minefield and then at time 31945 the AUV exited the minefield (Figure 23). The AUV then conducted its two ninety degrees turns at approximately time 32000 (Figure 23). The AUV then reentered the minefield at time 32100 and exited the minefield again at time 32140.

During early models the forward velocity was unbounded, upon initial deployment of the AUV when speeds were less than 1.0 meters per second the model accurately tracked velocity in the forward direction with an error of 0.06 meters per second (Figure 34).

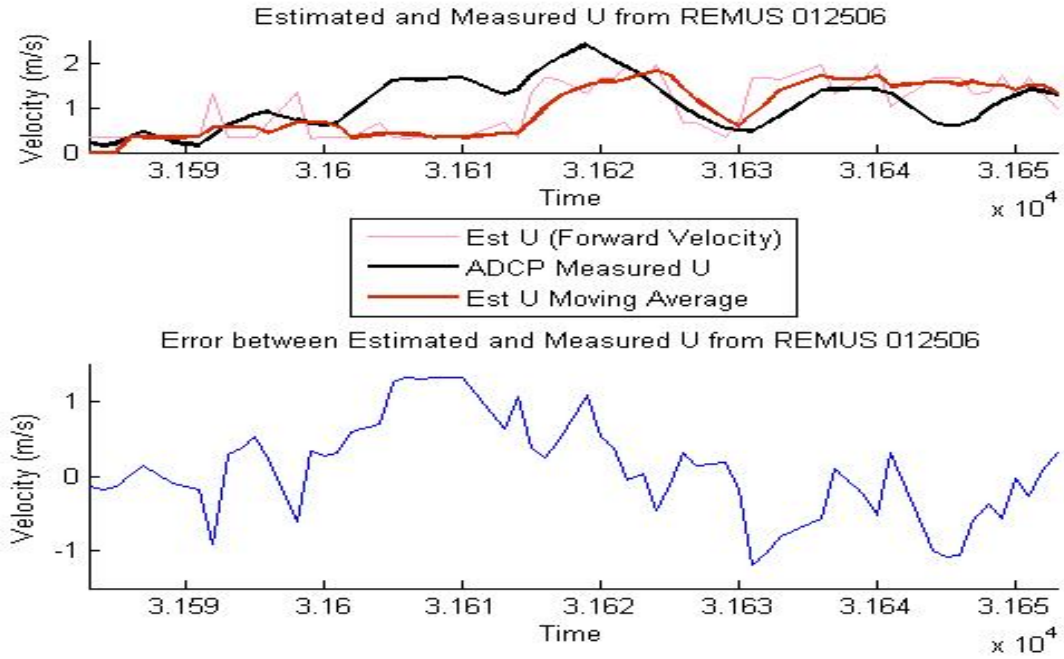

Figure 34. Unbounded Estimated and Measured U for Initial Deployment REMUS 012506

The average magnitude of the lateral velocity measured by the ADCP was 0.0844 meters per second for the duration of the entire mission. The lateral velocity estimates from the correlation coefficient were compared with the lateral velocity measured by the ADCP when the sonar images were recorded. The average of the lateral velocity measured by the ADCP was 0.0039 meters per second, while the lateral velocity never exceeded 0.5 meters per second throughout the entire mission. However, the lateral velocity estimates were zero through the entire REMUS mission. This is due to the relatively small motion in the lateral direction.

The highest magnitude of lateral velocity is expected during the turns of the vehicle. This is due to the characteristics of the vehicle which result in advance and transfer. The average magnitude of the lateral velocity measured by the ADCP DVL

during vehicle turns is 0.0936 meters per second. The angular accuracy of the FLS is 1.2 degrees, which at a range of 10 meters results in a lateral distance of 0.209 meters, and at a range of 90 meters is a lateral distance of 1.885 meters. Even at the shorter range the angular accuracy results in a distance which is much greater than the distances attempted to be measured in estimating the lateral velocity. It had been previously identified that sonar characteristically has poor angular accuracy (Kaylan, 2004), and previous work had concluded that sonar needed to be combined with and underwater camera to detect lateral motion.

The average error between the compass measured and estimated heading rates was 0.0844 degrees per second. However, within these results it was observed that the heading rate estimates did not track through the REMUS turns. The REMUS was preprogrammed to conduct quick ninety degree turns. During the turns on average only three FLS images were recorded. It is possible that the high speed of the turns resulted in insufficient information contained within the FLS field of view to conduct adequate correlation matching.

2. Lateral Velocity and Heading Rate Experiments 022307

To further examine the lateral velocity and heading rate inaccuracies additional experiments were conducted. The experiments were conducted in Monterey Harbor, California on 23 February 2007. The FLS was removed from the REMUS AUV, and was attached to a pole and lowered into the water from a pier. The FLS was walked laterally along the pier for several experiments; however the lateral velocity estimated was still consistently zero.

With a stop watch the FLS was also rotated by hand at a rate of one 360 degree revolution per minute (6 degrees per second) to simulate the REMUS AUV turning during a mission. The rotations were conducted in both the clockwise and counter-clockwise directions. The results of the experiment demonstrate that it is possible to accurately estimate the heading rate of the AUV. The average counter-clockwise heading

rate was estimated at 5.91 degrees per second, and the clockwise heading rate was estimated at -6.11 degrees per second. (Appendix A)

One of the heading rate experiments can be seen in Figure 35. In Figure 35 the sonar is initially lowered into the water from the dock, the initial noise is due to the sonar nose cone filling with water. Then the sonar was rotated counter-clockwise to align it to an initial orientation with the dock. Then the sonar was rotated by hand, in alternating directions, one complete 360 degree turn. The rotations can be seen starting clockwise then followed by counter-clockwise. A rotational rate of 6 degrees per second was attempted. The fluctuations in heading rate are believed to be repositioning of hands on the pole during the rotation of the sonar.

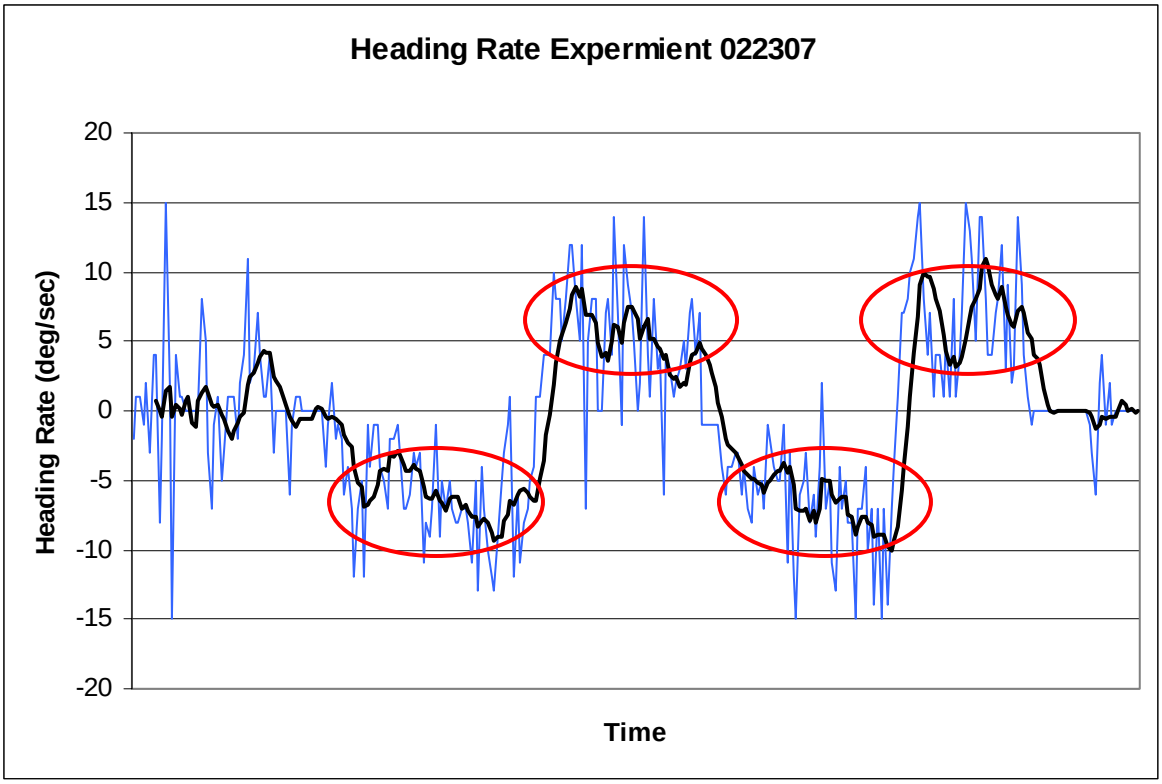

Figure 35. Simulated Heading Rate for REMUS 022307

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

This study is the first attempt to investigate the performance of the Blazed Array sonar system as a source of velocity inputs. This research demonstrated that it is possible to use the Blazed Array forward looking sonar images and the developed correlation coefficient based template matching techniques to estimate the forward velocity and heading rate of autonomous undersea vehicles. It was shown that the forward velocity of the AUV can be estimated to within the sonar resolution capabilities. In the lowest sonar resolution a two percent error in the forward velocity as compared to the ADCP DVL, whose accuracy is 0.1 centimeters per second, was achieved in this research. Utilizing higher Blazed Array sonar resolution settings the error should be reduced. These results demonstrate that the technique is useful and accurate even when the vehicle navigated over a relatively smooth ocean floor with no strong features in the FLS images.

This research also demonstrated in early tests that the velocity in the lateral direction was not estimated accurately with the FLS in the lowest resolution settings. Additional experimentation and previous research suggests that it may not be possible to track the lateral velocity using the developed methods applied to the FLS. This may be a result of the FLS in the lowest resolution settings, the small motion in the lateral direction, and the characteristically poor angular accuracy of the sonar (Kaylan, 2004).

B. RECOMMENDATIONS FOR FUTURE WORK

This thesis introduces a new method which is capable of accurate estimations for forward velocity and heading rates of autonomous vehicles in the undersea environment. Based on the results of the experimentation conducted for this thesis, additional missions should be planned to collect more data. Follow-on mission should incorporate slower turn rates and environments with varying bottom types. Utilizing the FLS and the template matching technique a mosaic of the ocean bottom could also be constructed.

The near-field effects also require additional evaluation. The exact source of the effects needs to be determined. During the mission the effects fluctuated and changed in size, possibly with the pitch, roll, and/or yaw of the vehicle. The effects could potentially contain useful information, which could be gained through modeling the effects.

Significant research is required to resolve the questions associated with the lateral velocity. Follow-on missions should be conducted with the sonar in higher resolution settings determine if accurate results could be achieved with the increased resolution. There are potential methods to work around the issues associated with the poor angular accuracy of the sonar. One method would be applying this imagery based forward velocity estimation to the REMUS AUV side scan sonars. Estimating the forward velocity from the side scan sonars could result in accurate estimates of the vehicles lateral velocity.

For this method to be useful it must be employed in real-time onboard the AUVs. There are several methods which could modify this program and reduce the computational time for the velocity estimates. The current search conducted in Matlab® is time expensive, experimentation with various optimization methods could result in real-time velocity estimates.

Once a real-time velocity estimate can be provided and employed in the AUVs navigation and position estimation systems, these inputs could be used vice the ADCP measurements. Evaluating the performance of the vehicle utilizing these estimates vice measurement from the ADCP DVL would be necessary. Eliminating the RDI Doppler as an onboard sensor has the following benefits:

- Electrical load reduced, increasing mission endurance.
- Reduced electrical noise impacts on Forward Look Sonar and Side Scan Sonar, improved obstacle avoidance and mine hunting performance.
- Reduced capital cost. $35k (RDI Doppler)
- Increases payload space for other components or reduces overall size of AUV.
- Enhanced mission flexibility due to above.

There is future work in incorporating this technique and other computer vision techniques to the obstacle avoidance and simultaneous localization and mapping problems. Computer vision processes applied in non-traditional roles can further enhance the capabilities of autonomous underwater vehicles. However these processes could be incorporated in a multitude of unmanned vehicles, to include ground, surface and aerial systems.

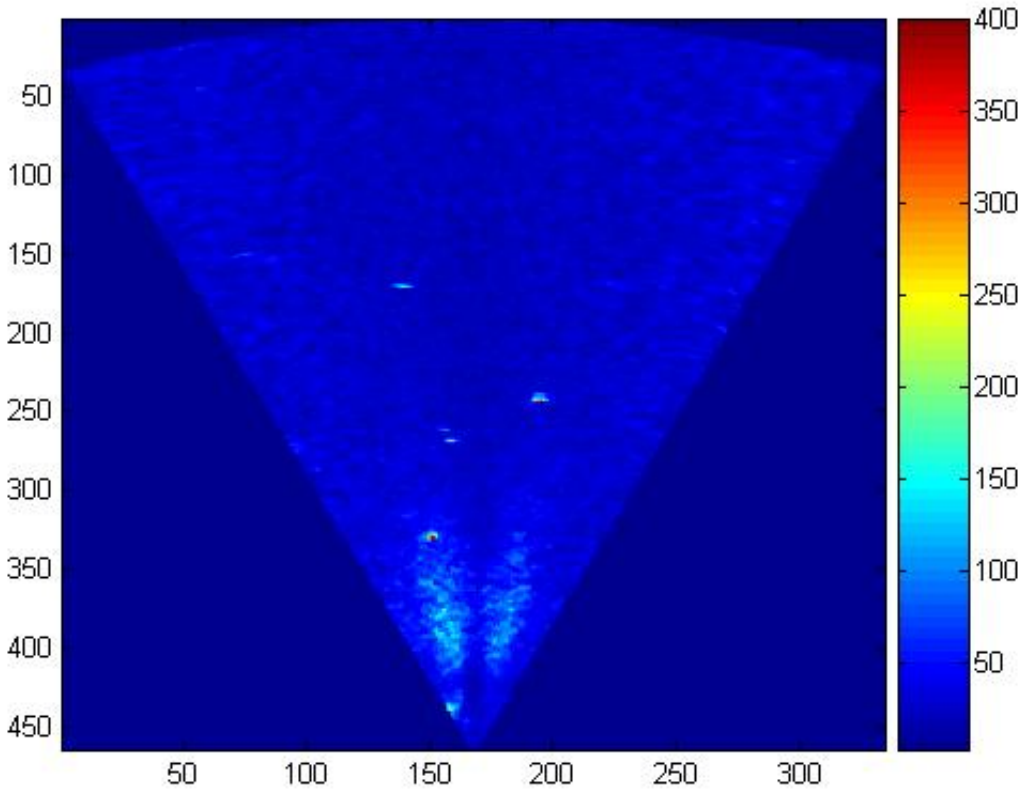

Figure 36. Blazed Array FLS Image of Multiple Features REMUS AUV 012506

One of the current challenges presently being worked on by many robotic communities is the simultaneous localization and mapping (SLAM) problem (Leonard, 2003). One of the critical components for an unmanned vehicle to map an unknown environment is its ability to locate itself on a partially explored map, or unknown environment. The AUV having been placed in an unknown location in an unknown

environment would create a map, using only relative observations of features, and simultaneously use it to navigate. This ability would remove the need for the vehicle to have *a priori* knowledge of the environment. The observed relative position of the feature is added to the map and used to update the relative position of the vehicle in the local map. The ability of an unmanned vehicle to travel through an unknown environment and map features currently has several challenges. Errors associated with the model of the vehicles motion and errors associated with the sensors both contribute to inaccuracies in feature and vehicle positions within the map. Another challenge is the data association problem. This is the robot's inability to determine whether a feature that it is currently detecting is the same feature that it had previously detected. Without the ability to associate the data correctly an accurate map can not be produced and the unmanned vehicle may become lost in the unknown environment.

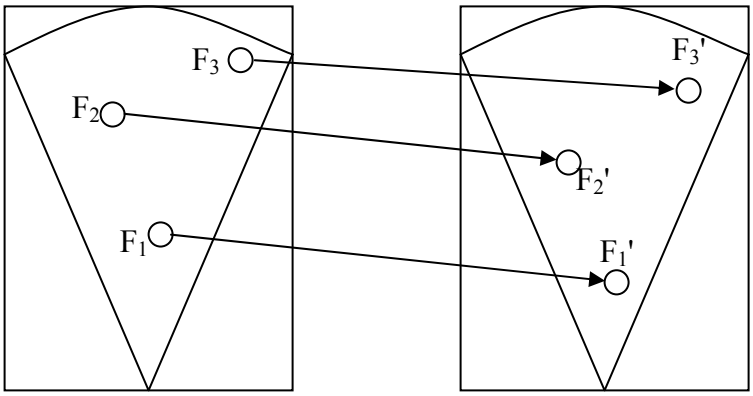

Figure 37. Geometric Relationship between Multiple Features

Sonar is one method that is used to measure the relative position of features in an undersea environment. Feature detectors are used to extract the features from the sonar images and the relative positions, range and bearing can be determined. The relative positions of the features are used in position estimation filters, such as an Extended

Kalman Filter (EKF), to determine the updated position of the unmanned vehicle in the environment. However, their use in the EKF requires the knowledge of the specific feature and its location (Figure 36). This can prevent confusing the features and sending bad inputs into the EKF, which would then result in erroneous estimates of the vehicles position in the environment. Using velocity estimates from the sonar images we could accurately predict the location of features from one image to the next.

$$\begin{bmatrix} F_1' \\ F_2' \\ F_3' \end{bmatrix} = \mathrm{R} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} + \mathrm{T} + \varepsilon \tag{44}$$

In other words, the geometrical relationship of multiple objects in a sequence of sonar images should hold. Errors associated with the relationships would be a result of the sensor model, and the sonar imaging. By comparing the affine transformation from the image correlation technique to the movement from the individual features detected there should be close to a one to one mapping (Figure 37). Over a sequence of images this comparison should be useful in detecting additional features, such as features which were previously undetected or features which may have moved since the original mapping. Using this prediction, the estimated position can be compared to the measured position, to ensure that individual features are tracked accordingly. Ensuring that the new positions of the features are input correctly would result in a more accurate vehicle position update.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A: SIMULATION RESULTS

The average error through the entire mission between the ADCP measured and imagery-based estimated forward velocities was 0.0392 meters per second, which results in approximately 2 percent error in the forward velocity estimate compared to the ADCP DVL measurements. The average magnitude of the error for the entire mission is 0.2291 meters per second, which is approximately the resolution of the sonar which is 0.1940 meters per pixel.

REMUS AUV Mission 012506

	Start Time	Stop Time	U Average Error	U Average Absolute Error	Heading Rate Average Error	HR Average Absolute Error
	31627	31849	-0.0758	0.2566	0.3863	0.7603
	31850	32063	-0.0627	0.1996	-0.8157	1.2354
	32064	32279	-0.0805	0.1879	0.7774	1.1738
	32280	32491	-0.0464	0.1870	-0.9055	1.3176
	32492	32710	-0.0432	0.2068	0.7299	1.1886
	32711	32919	-0.0372	0.1820	-0.8257	1.3478
	32920	33129	-0.0380	0.2499	1.3502	2.1270
	33130	33344	-0.0168	0.2502	-0.6336	1.4846
	33345	33559	0.0253	0.2975	0.7953	1.2324
	33561	33779	-0.0675	0.2511	-0.9357	1.2810
	33780	33996	-0.0141	0.2625	0.8199	1.1594
	33997	34229	-0.0139	0.2184	0.2703	0.6211
Entire Mission	31627	34229	-0.0392	0.2291	0.0844	1.2441

Heading Rate Experiment 022307	
Clockwise Rotation (deg/sec)	-5.67
Counter-Clockwise Rotation (deg/sec)	5.37
Clockwise Rotation (deg/sec)	-6.57
Counter-Clockwise Rotation (deg/sec)	6.46
Clockwise Rotation (deg/sec)	-6.09
Average Clockwise Rotation (deg/sec)	-6.11
Average Counter-Clockwise Rotation (deg/sec)	5.91

Rotational Rates Attempted (deg/sec)	+/- 6.00

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: MATLAB CODE FOR VELOCITY ESTIMATES

The following MATLAB® code was used for running simulations on the data sets. The original code was developed by M. Dolbec. The codes contained in Appendices B and C are intended to run in MATLAB® with a specific data set structure from the REMUS vehicle. These sonar images and data sets may be obtained from the NPS Center for AUV Research or the codes can be tailored to run with specific data sets to provide the correct velocity estimates.

```
% Correlation Filter between sequential images
% Mike Dolbec       Last modified 01FEB07

% Methodology   ************************
%
% remove the near field effects
% Apply heading rate and velocity estimate to sonar image (rotation and
translation)
% utilize nested for loops to determine highest correlation coefficient
% then motion analysis for combined u and HR
clc,

load state_lbl_adcp_pings012506_01.mat

% >> REMUS(k)  k = 1:11805

%           lblLatitude: -999
%          lblLongitude: -999
%          adcpLatitude: 36.7169
%         adcpLongitude: -121.8202
%       forwardVelocity: -0.1359
%     starboardVelocity: 0.0689
%      verticalVelocity: -0.0620
%              altitude: 19.1839
%              latitude: -999
%             longitude: -999
%                 depth: -999
%        compassHeading: -999
%           headingRate: -999
%     estimatedVelocity: -999
%                 pitch: -999
%             pitchRate: -999
%                  roll: -999
%              rollRate: -999
%         flsFileNumber: -999
%                  time: 3.1583e+004
```

```
Data = [];
Time = [];
Fig  = [];
RTrack = [];
MAXtime = 0;

% Initial Velocity Estimates
vel = 1.0;
Delta_head = 0;

j = 1;     % lower limit of the images to process
k = 11805;     % upper limit of the images to process

i=1;
%%
% determine Heading Rate and Velocity
for n = j+1:k+1

    if (REMUS(n-1).compassHeading) > -999
        Data(1,i) = REMUS(n-1).compassHeading;
        Data(5,i) = REMUS(n-1).estimatedVelocity;
        Data(7,i) = REMUS(n-1).headingRate;
        Data(9,i) = REMUS(n-1).pitch;
        Data(11,i) = REMUS(n-1).roll;
        Time(3,i) = REMUS(n-1).time;
        ind0 = n-1;

        if (REMUS(n-1).flsFileNumber) == -999
                for m = n+1:k+1
                    if (REMUS(m).flsFileNumber) > -999
                        Data(3,i) = REMUS(m).flsFileNumber;
                        Time(1,i) = REMUS(m).time;
                        ind1 = m;
                        break
                    end
                end
            end

            if (REMUS(n).compassHeading) == -999
                for m = n+1:k+1
                    if (REMUS(m).compassHeading) > -999
                        Data(2,i) = REMUS(m).compassHeading;
                        Data(6,i) = REMUS(m).estimatedVelocity;
                        Data(8,i) = REMUS(m).headingRate;
                        Data(10,i) = REMUS(m).pitch;
                        Data(12,i) = REMUS(m).roll;
                        Time(4,i) = REMUS(m).time;
                        ind2 = m;
                        break
                    end
                end
            end

            if (REMUS(ind1+1).flsFileNumber) == -999
```

```
                for l = ind1+1:k+1
                    if (REMUS(l).flsFileNumber) > -999
                        Data(4,i) = REMUS(l).flsFileNumber;
                        Time(2,i) = REMUS(l).time;
                        ind3 = l;
                        break
                    end
                end
            end

        if Data(3,1) < Data(4,1)

        Data;
        Time;
        dt = Time(2,1)-Time(1,1); % time since the AUV was deployed
        avgT = (Time(3,1)+Time(4,1)) / 2;
        vel1 = (Data(5,1)+Data(6,1)) / 2;
        pitch = (Data(9,1)+Data(10,1)) / 2;
        roll = (Data(11,1)+Data(12,1)) / 2;
%%
%*********************************************************************
%       Need to handle the discontinuity jumps between 0 and 359
%*********************************************************************

        if abs(Data(2,1)-Data(1,1)) <= 180
            Delta_head1 = Data(2,1)-Data(1,1);
        end

        if Data(2,1)-Data(1,1) > 180
            Delta_head1 = Data(2,1)-Data(1,1) - 360;
        end

        if Data(1,1)-Data(2,1) > 180
            Delta_head1 = Data(2,1)-Data(1,1) + 360;
        end

        head_rate = (Delta_head1)/dt;
        head_rate1 = (Data(7,1)+Data(8,1)) / 2;

%%
%*********************************************************************
%       Retrieves the appropiate sonar images
%*********************************************************************

tic

for vel = 1.0:0.1:2.0

        IM1 = OpenSonarImage(Data(3,1));
        IM2 = OpenSonarImage(Data(4,1));

[rows cols] = size (IM1);
```

```
%*************************************************************************
%       Determine threshold values
%*************************************************************************

Thresh = IM1(17:220, 80:260);
S = mean(Thresh);
threshold1 = mean(S);

Thresh = IM2(17:220, 80:260);
S = mean(Thresh);
threshold2 = mean(S);

%*************************************************************************
%       Applys a threshold to remove the Near-Field Effects
%*************************************************************************

        threshold = 50;     % feature detection intensity threshold

        for X = round(rows/2):rows
            for Y = 1:cols
                if IM1(X,Y) >= threshold;
                    IM1(X,Y) = threshold1;
                end
            end
        end

        for X = round(rows/2):rows
            for Y = 1:cols
                if IM2(X,Y) >= threshold;
                    IM2(X,Y) = threshold2;
                end
            end
        end

%*************************************************************************
%    Threshold to determine what is being tracked in the sonar images
%*************************************************************************
Max = max(max(IM1));

%         threshold = 40;     % feature detection intensity threshold
%
%         for X = 1:rows
%             for Y = 1:cols
%                 if IM1(X,Y) >= threshold;
%                     IM1(X,Y) = threshold1;
%                 end
%             end
%         end
%
%         for X = 1:rows
%             for Y = 1:cols
%                 if IM2(X,Y) >= threshold;
%                     IM2(X,Y) = threshold2;
%                 end
```

```
%             end
%         end

%*************************************************************************
%                   End Preprocessing
%*************************************************************************

%         dt = 2;
        pixelsPerMeter = rows / 90;  % Range (m) of the Blazed Array
FLS
        PixelVelx = round (vel * pixelsPerMeter * dt);

        x = 0;

% Image_mod3 = IM1(1:rows-(PixelVelx+x), 1:cols);
% Actual_image3 = IM2((PixelVelx+x)+1:rows, 1:cols);
%
% r = corr2(Image_mod3, Actual_image3)

%*************************************************************************
%        Determines the Origin of the image
%*************************************************************************

            % determines the origin of the image
            [rows1 cols1] = size (IM1);

            for r = round(rows1/2):rows1;
                if sum(IM1(r,:)) == 0
                    origin_rows1 = (r-1);
                    X1 = origin_rows1 - (PixelVelx+x);
                    break
                end
            end

            for c = 1:cols1
                if (IM1(origin_rows1,c)) > 0
                    Y1 = (c+1);
                    break
                end
            end

%*************************************************************************
%        Determines the points of the image to be excluded IM1
%*************************************************************************

% this determines the point from which the line will be drawn
% this will exclude non-applicable portions due to U

% X2 = 33-(PixelVelx+x)
%
% Y2 = 1;

[rows1 cols1] = size (IM1);
```

```
Y2 = 1;
% Y2 = cols1

for q = 1:rows1
    if IM1(q,Y2) > 0
        X2 = (q+1)-(PixelVelx+x);
        X2a = (q+1);
        break
    end
end

%**************************************************************************
%        Create the line from the two points
%**************************************************************************

IM1 = linept(IM1, X1, Y1, X2, Y2);

% figure(1),
% subplot(2,2,1)
% imagesc(IM1)

%**************************************************************************
%        Remove the non-applicable portion of the image IM1
%**************************************************************************

threshold = 0;  % value of non-image pixels
[rows cols] = size(IM1);

for X = 1:rows
    for Y = 1:cols
        if IM1(X,Y) >= threshold;
            IM1(X,Y) = 0;
        elseif IM1(X,Y) == -1;
            IM1(X,Y) = 0;
            break
        end
    end
end

% figure(1),
% subplot(2,2,2)
% imagesc(IM1)

%**************************************************************************
%        Determines the point of the image to be excluded IM2
%**************************************************************************

% this determines the point from which the line will be drawn
% this will exclude non-applicable portions due to U

% X2 = 33-(PixelVelx+x);
%
```

```
% Y2 = cols;

[rows1 cols1] = size (IM1);

% Y2 = 1;
Y2 = cols1;

for q = 1:rows1
    if IM1(q,Y2) > 0
        X2 = (q+1)-(PixelVelx+x);
        break
    end
end

%*********************************************************************
%        Create the line from the two points for U
%*********************************************************************

IM1a = linept(IM1, X1, Y1, X2, Y2);
% figure(1),
% subplot(2,2,3)
% imagesc(IM1a)

%*********************************************************************
%        Remove the non-applicable portion of the image IM2
%*********************************************************************

threshold = -1;  % value of non-image pixels
[rows cols] = size(IM1a);

% for h = 1:X2
%     for Y = 1:cols
%         IM1a(h,Y) = 0;
%     end
% end

for X = 1:rows
    for Y = 1:cols
        if IM1a(X,Y) == -1;
            for g = Y:cols
                IM1a(X,g) = 0;
            end
        end
    end
end

% figure(1),
% subplot(2,2,4)
% imagesc(IM1a)

%*********************************************************************
%        Determines the Origin of the 2 images (initial estimates)
%*********************************************************************
```

```
            % determines the origin of the unrotated image
            [rows1 cols1] = size (IM2);

            for r = round(rows1/2):rows1
                if sum(IM2(r,:)) == 0
                    origin_rows2a = (r-1);
                    break
                end
            end

            for c = 1:cols1
                if (IM2(origin_rows2a,c)) > 0
                    origin_cols2a = (c);
                    break
                end
            end


            % determines the origin of the image
%             [rows1a cols1a] = size (IM1a);
%
%             for r = round(rows1/2):rows1a
%                 if sum(IM1a(r,:)) == 0
%                     origin_rows1a = (r-1);
%                     break
%                 end
%             end
%
%             for c = 1:cols1a
%                 if (IM1a(origin_rows1a,c)) > 0
%                     origin_cols1a = (c);
%                     break
%                 end
%             end

            origin_rows1a = X1;
            origin_cols1a = Y1;

%**********************************************************************
%    Addition of zeros to match origins (initial estimates)
%**********************************************************************

% to calculate the correlation coefficient the origins of the images
% must be at the same pixel location to ensure a truthful comparison

        [rows1 cols1] = size (IM2);
        [rows1a cols1a] = size (IM1a);

            if origin_rows1a-origin_rows2a == 0
                IM2 = IM2;
                Extra_Rows = 0;
            elseif origin_rows1a-origin_rows2a > 0
```

```
            Extra_Rows = origin_rows1a-origin_rows2a;
            bottom = zeros(Extra_Rows,cols1);
            IM2 = [IM2; bottom];
        elseif origin_rows2a-origin_rows1a > 0
            Extra_Rows = origin_rows2a-origin_rows1a;
            top = zeros(Extra_Rows,cols1);
            IM1a = [top; IM1a];
            IM2 = IM2;
        end


        if origin_cols1a-origin_cols2a == 0
            Extra_Cols = 0;
        elseif origin_cols1a-origin_cols2a > 0
            Extra_Cols = origin_cols1a-origin_cols2a;
            left = zeros(rows1,Extra_Cols);
            IM2 = [left, IM2];
        else
            Extra_Cols = origin_cols2a-origin_cols1a;
            left = zeros(rows1a,Extra_Cols);
            IM1a = [left, IM1a];
        end

%**************************************************************************
%        Check the Origin of the 2 images (initial estimates)
%**************************************************************************

% this is to ensure the code is adjusting the images properly

            % determines the origin of the image
            [rows1 cols1] = size (IM2);

            for r = round(rows1/2):rows1
                if sum(IM2(r,:)) == 0
                    origin_rows2a = (r-1);
                    break
                end
            end

            for c = 1:cols1
                if (IM2(origin_rows2a,c)) > 0
                    origin_cols2a = (c);
                    break
                end
            end

            % determines the origin of the image
            [rows1a cols1a] = size (IM1a);

            for r = round(rows1/2):rows1a
                if sum(IM1a(r,:)) == 0
                    origin_rows1a = (r-1);
                    break
                end
```

```
            end

            for c = 1:cols1a
                if (IM1a(origin_rows1a,c)) > 0
                    origin_cols1a = (c);
                    break
                end
            end

%*********************************************************************
%     Addition of zeros to match matrix sizes (initial estimates)
%*********************************************************************

% to calculate the correlation coefficient the images (matrices)
% must be of identical sizes

            [rows1a cols1a] = size(IM1a);
            [rows1b cols1b] = size(IM2);

            if rows1a-rows1b > 0
                Extra_Rows = rows1a-rows1b;
                bottom = zeros(Extra_Rows,cols1b);
                IM2 = [IM2; bottom];
            else
                Extra_Rows = rows1b-rows1a;
                bottom = zeros(Extra_Rows,cols1a);
                IM1a = [IM1a; bottom];
            end

            [rows1a cols1a] = size(IM1a);
            [rows1b cols1b] = size(IM2);

            if cols1a-cols1b > 0
                Extra_Cols = cols1a-cols1b;
                right = zeros(rows1b,Extra_Cols);
                IM2 = [IM2, right];
            else
                Extra_Cols = cols1b-cols1a;
                right = zeros(rows1a,Extra_Cols);
                IM1a = [IM1a, right];
            end

%*********************************************************************
%        Determines the upper left corner of the two images
%*********************************************************************

            [rows2 cols2] = size (IM2);

            for c = 1:cols2
                if sum(IM2(:,c)) > 0
                    left2a_col = (c);
                    break
                end
```

```
            end

            for r = 1:rows2
                if (IM2(r,left2a_col)) > 0
                    left2a_row = (r);
                    break
                end
            end

            [rows1 cols1] = size (IM1a);

            for c = 1:cols1
                if sum(IM1a(:,c)) > 0
                    left1a_col = (c);
                    break
                end
            end

            for r = 1:rows1
                if (IM1a(r,left1a_col)) > 0
                    left1a_row = (r);
                    break
                end
            end

%**************************************************************************
%        removes overlap which would reduce the correlation
%**************************************************************************

% this will exclude non-applicable portions due to U

            if left1a_row - left2a_row > 0

                for h = 1:left1a_row
                    for Y = 1:cols2
                        IM2(h,Y) = 0;
                    end
                end
                for h = 1:left1a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end

            else

                for h = 1:left2a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end
                for h = 1:left2a_row
                    for Y = 1:cols2
```

```
                        IM2(h,Y) = 0;
                    end
                end
            end

% Image_mod3 = IM1a;
% Actual_image3 = IM2;
%
% r = corr2(Image_mod3, Actual_image3)

Image_mod3 = IM1a;
Actual_image3 = IM2;

%*********************************************************************
%     Determines the correlation coefficient, and tracks the highest
%*********************************************************************

r = corr2(Image_mod3, Actual_image3);

    if vel == 1.0

            u = ((PixelVelx + x) / pixelsPerMeter) / dt;
            HR = (0);
            rtrack = [r; u; vel1;
                      HR; head_rate; head_rate1;
                      avgT; pitch; roll];

    else

        if r > rtrack(1,1)
            u = ((PixelVelx + x) / pixelsPerMeter) / dt;
            HR = (0);
            rtrack = [r; u; vel1;
                      HR; head_rate; head_rate1;
                      avgT; pitch; roll];
        end

    end
% RTrack_plot = [RTrack_plot, rtrack];

IM1u = IM1a;
IM1o = IM2;



%%
%*********************************************************************
%        Performs the operation for lateral velocity
%*********************************************************************

horizontal_angle1 = 0;

% Determine the angle
```

```
for  y = 1:1:15

    horizontal_angle = horizontal_angle1 + y;
%        horizontal_radian = (dt * estimated_v) * pi / 180;

%*********************************************************************
%        Determines the Origin of the image
%*********************************************************************

% figure(2),
% subplot(3,2,1)
% imagesc(IM1u);
%
% figure(2),
% subplot(3,2,2)
% imagesc(IM1o);

[rows1 cols1] = size(IM1o);

for r = round(rows1/2):rows1;
    if sum(IM1o(r,:)) == 0
        origin_rows1 = (r-1);
        X1 = origin_rows1;
        break
    end
end

for c = 1:cols1
    if (IM1o(origin_rows1,c)) > 0
        Y1 = (c);
        break
    end
end

%*********************************************************************
%        Determines the point of the image to be excluded IM1
%*********************************************************************

X2 = X2a;

% since 25 degrees equals 167 pixels
% 1 degree = 6.68 pixels ~= 7 pixels

Y2 = round(horizontal_angle * 7.42);

%*********************************************************************
%        Create the line from the two points
%*********************************************************************

IM1a = linept(IM1o, X1, Y1, X2, Y2);
% figure(2),
% subplot(3,2,4)
% imagesc(IM1a)
```

```
%*********************************************************************
%        Remove the non-applicable portion of the image IM1
%*********************************************************************

threshold = 0;  % value of non-image pixels
[rows cols] = size(IM1a);

for X = 1:rows
    for Y = 1:cols
        if IM1a(X,Y) >= threshold;
            IM1a(X,Y) = 0;
        elseif IM1a(X,Y) == -1;
            IM1a(X,Y) = 0;
            break
        end
    end
end

% figure(2),
% subplot(3,2,6)
% imagesc(IM1a)


%*********************************************************************
%        Determines the point of the image to be excluded IM2
%*********************************************************************

% determines the origin of the image
[rows1 cols1] = size (IM1u);

for r = round(rows1/2):rows1;
    if sum(IM1u(r,:)) == 0
        origin_rows1 = (r-1);
        X1 = origin_rows1;
        break
    end
end

for c = 1:cols1
    if (IM1u(origin_rows1,c)) > 0
        Y1 = (c);
        break
    end
end


X2 = X2a;

% since 25 degrees equals 167 pixels
% 1 degree = 6.68 pixels ~= 7 pixels

Y2 = cols1-round(horizontal_angle * 7.42);
```

```
%*********************************************************************
%        Create the line from the two points IM2
%*********************************************************************

IM2a = linept(IM1u, X1, Y1, X2, Y2);
% figure(2),
% subplot(3,2,3)
% imagesc(IM2a)

%*********************************************************************
%        Remove the non-applicable portion of the image IM2
%*********************************************************************

threshold = -1;  % value of non-image pixels
[rows cols] = size(IM2a);

for h = 1:X2
    for Y = 1:cols
        IM2a(h,Y) = 0;
    end
end

for X = 1:rows
    for Y = 1:cols
        if IM2a(X,Y) == -1;
            for g = Y:cols
                IM2a(X,g) = 0;
            end
        end
    end
end

% figure(2),
% subplot(3,2,5)
% imagesc(IM2a)

IM1a = imrotate(IM1a, (horizontal_angle), 'loose');
% figure(3), imagesc(IM1a)



%*********************************************************************
%        Determines the Origin of the 2 images (initial estimates)
%*********************************************************************

            % determines the origin of the unrotated image
            [rows1 cols1] = size (IM2a);

            for r = round(rows1/2):rows1
                if sum(IM2a(r,:)) == 0
                    origin_rows2a = (r-1);
                    break
```

```
            end
        end

        for c = 1:cols1
            if (IM2a(origin_rows2a,c)) > 0
                origin_cols2a = (c);
                break
            end
        end

        % determines the origin of the left rotated image
        [rows1a cols1a] = size (IM1a);

        for r = round(rows1/2):rows1a
            if sum(IM1a(r,:)) == 0
                origin_rows1a = (r-1);
                break
            end
        end

        for c = 1:cols1a
            if (IM1a(origin_rows1a,c)) > 0
                origin_cols1a = (c);
                break
            end
        end

%**********************************************************************
%    Addition of zeros to match origins (initial estimates)
%**********************************************************************

        if origin_rows1a-origin_rows2a == 0
            IM2a = IM2a;
            Extra_Rows = 0;
        elseif origin_rows1a-origin_rows2a > 0
            Extra_Rows = origin_rows1a-origin_rows2a;
            top = zeros(Extra_Rows,cols1);
            IM2a = [top; IM2a];
        else
            Extra_Rows = origin_rows2a-origin_rows1a;
            bottom = zeros(Extra_Rows,cols1a);
            IM1a = [IM1a; bottom];
            IM2a = IM2a;
        end

    [rows1 cols1] = size (IM2a);
    [rows1a cols1a] = size (IM1a);

        if origin_cols1a-origin_cols2a == 0
            Extra_Cols = 0;
        elseif origin_cols1a-origin_cols2a > 0
            Extra_Cols = origin_cols1a-origin_cols2a;
            left = zeros(rows1,Extra_Cols);
            IM2a = [left, IM2a];
```

```
            else
                Extra_Cols = origin_cols2a-origin_cols1a;
                left = zeros(rows1a,Extra_Cols);
                IM1a = [left, IM1a];
            end

%*********************************************************************
%     Addition of zeros to match matrix sizes (initial estimates)
%*********************************************************************

            [rows1a cols1a] = size(IM1a);
            [rows1b cols1b] = size(IM2a);

            if rows1a-rows1b > 0
                Extra_Rows = rows1a-rows1b;
                bottom = zeros(Extra_Rows,cols1b);
                IM2a = [IM2a; bottom];
            else
                Extra_Rows = rows1b-rows1a;
                bottom = zeros(Extra_Rows,cols1a);
                IM1a = [IM1a; bottom];
            end

        [rows1a cols1a] = size(IM1a);
        [rows1b cols1b] = size(IM2a);

            if cols1a-cols1b > 0
                Extra_Cols = cols1a-cols1b;
                right = zeros(rows1b,Extra_Cols);
                IM2a = [IM2a, right];
            else
                Extra_Cols = cols1b-cols1a;
                right = zeros(rows1a,Extra_Cols);
                IM1a = [IM1a, right];
            end
% figure(4),
% subplot(2,1,1)
% imagesc(IM1a)
% subplot(2,1,2)
% imagesc(IM2a)

% Image_mod3 = IM1a;
% Actual_image3 = IM2a;
%
% r = corr2(Image_mod3, Actual_image3)



%*********************************************************************
%        Determines the upper left corner of the two images
%*********************************************************************

if  horizontal_angle >= 0
```

```
            [rows2 cols2] = size (IM2a);

            for c = 1:cols2
                if sum(IM2a(:,c)) > 0
                    left2a_col = (c);
                    break
                end
            end

            for r = 1:rows2
                if (IM2a(r,left2a_col)) > 0
                    left2a_row = (r);
                    break
                end
            end

            [rows1 cols1] = size (IM1a);

            for c = 1:cols1
                if sum(IM1a(:,c)) > 0
                    left1a_col = (c);
                    break
                end
            end

            for r = 1:rows1
                if (IM1a(r,left1a_col)) > 0
                    left1a_row = (r);
                    break
                end
            end

%**************************************************************************
%        removes overlap which would reduce the correlation
%**************************************************************************

            if left1a_row - left2a_row > 0

                for h = 1:left1a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
                for h = 1:left1a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end

            else

                for h = 1:left2a_row
```

```
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end
                for h = 1:left2a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
            end

% figure(5),
% subplot(2,1,1)
% imagesc(IM1a)
% subplot(2,1,2)
% imagesc(IM2a)

Else

%*********************************************************************
%        Determines the upper right corner of the two images
%*********************************************************************

            [rows2 cols2] = size (IM2a);

            for c = round(cols2/2):cols2
                if sum(IM2a(:,c)) > 0
                    right2a_col = (c-1);
                    break
                end
            end

            for r = 1:rows2
                if (IM2a(r,right2a_col)) > 0
                    right2a_row = (r);
                    break
                end
            end

            [rows1 cols1] = size (IM1a);

            for c = round(cols1/2):cols1
                if sum(IM1a(:,c)) > 0
                    right1a_col = (c-1);
                    break
                end
            end

            for r = 1:rows1
                if (IM1a(r,right1a_col)) > 0
                    right1a_row = (r);
                    break
                end
            end
```

```
%*********************************************************************
%        removes overlap which would reduce the correlation
%*********************************************************************

            if right1a_row - right2a_row > 0

                for h = 1:right1a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
                for h = 1:right1a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end

            else

                for h = 1:right2a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end
                for h = 1:right2a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
            end

% figure(5),
% subplot(2,1,1)
% imagesc(IM1a)
% subplot(2,1,2)
% imagesc(IM2a)

end

% figure(2),
% subplot(4,2,8)
% imagesc(IM1a)
% subplot(4,2,7)
% imagesc(IM2a)

Image_mod3 = IM1a;
Actual_image3 = IM2a;

%*********************************************************************
%     Determines the correlation coefficient, and tracks the highest
%*********************************************************************
```

```
r = corr2(Image_mod3, Actual_image3);

        if r > rtrack(1,1)
            u = ((PixelVelx + x) / pixelsPerMeter) / dt;
            HR = (horizontal_angle);
            rtrack = [r; u; vel1;
                      HR; head_rate; head_rate1;
                      avgT; pitch; roll];
        end

% RTrack_plot = [RTrack_plot, rtrack];

%%
    horizontal_angle = horizontal_angle1 - y;
%        horizontal_radian = (dt * estimated_v) * pi / 180;
%*********************************************************************
%        Determines the Origin of the image
%*********************************************************************

% figure(6),
% subplot(3,2,1)
% imagesc(IM1u);
%
% figure(6),
% subplot(3,2,2)
% imagesc(IM1o);

[rows1 cols1] = size(IM1o);

for r = round(rows1/2):rows1;
    if sum(IM1o(r,:)) == 0
        origin_rows1 = (r-1);
        X1 = origin_rows1;
        break
    end
end

for c = 1:cols1
    if (IM1o(origin_rows1,c)) > 0
        Y1 = (c);
        break
    end
end

%*********************************************************************
%        Determines the point of the image to be excluded IM1
%*********************************************************************

X2 = X2a;

% since 25 degrees equals 167 pixels
% 1 degree = 6.68 pixels ~= 7 pixels
```

```
Y2 = cols1+round(horizontal_angle * 7.42);

%**********************************************************************
%        Create the line from the two points
%**********************************************************************

IM1a = linept(IM1o, X1, Y1, X2, Y2);
% figure(6),
% subplot(3,2,4)
% imagesc(IM1a)

%**********************************************************************
%        Remove the non-applicable portion of the image IM1
%**********************************************************************

threshold = -1;  % value of non-image pixels
[rows cols] = size(IM1a);

for h = 1:X2
    for Y = 1:cols
        IM1a(h,Y) = 0;
    end
end

for X = 1:rows
    for Y = 1:cols
        if IM1a(X,Y) == -1;
            for g = Y:cols
                IM1a(X,g) = 0;
            end
        end
    end
end

% figure(6),
% subplot(3,2,6)
% imagesc(IM1a)
%**********************************************************************
%        Determines the point of the image to be excluded IM2
%**********************************************************************

% determines the origin of the image
[rows1 cols1] = size (IM1a);

for r = round(rows1/2):rows1;
    if sum(IM1u(r,:)) == 0
        origin_rows1 = (r-1);
        X1 = origin_rows1;
        break
    end
end

for c = 1:cols1
```

```
    if (IM1u(origin_rows1,c)) > 0
        Y1 = (c);
        break
    end
end


X2 = X2a;

% since 25 degrees equals 167 pixels
% 1 degree = 6.68 pixels ~= 7 pixels

Y2 = -round(horizontal_angle * 7.42);

%**********************************************************************
%        Create the line from the two points IM2
%**********************************************************************

IM2a = linept(IM1u, X1, Y1, X2, Y2);
% figure(6),
% subplot(3,2,3)
% imagesc(IM2a)

%**********************************************************************
%        Remove the non-applicable portion of the image IM2
%**********************************************************************

threshold = 0;  % value of non-image pixels
[rows cols] = size(IM2a);

for X = 1:rows
    for Y = 1:cols
        if IM2a(X,Y) >= threshold;
            IM2a(X,Y) = 0;
        elseif IM2a(X,Y) == -1;
            IM2a(X,Y) = 0;
            break
        end
    end
end

% figure(6),
% subplot(3,2,5)
% imagesc(IM2a)

IM1a = imrotate(IM1a, (horizontal_angle), 'loose');

%**********************************************************************
%        Determines the Origin of the 2 images (initial estimates)
%**********************************************************************

           % determines the origin of the unrotated image
           [rows1 cols1] = size (IM2a);
```

```
for r = round(rows1/2):rows1
    if sum(IM2a(r,:)) == 0
        origin_rows2a = (r-1);
        break
    end
end

for c = 1:cols1
    if (IM2a(origin_rows2a,c)) > 0
        origin_cols2a = (c);
        break
    end
end

% determines the origin of the left rotated image
[rows1a cols1a] = size (IM1a);

for r = round(rows1/2):rows1a
    if sum(IM1a(r,:)) == 0
        origin_rows1a = (r-1);
        break
    end
end

for c = 1:cols1a
    if (IM1a(origin_rows1a,c)) > 0
        origin_cols1a = (c-0);
        break
    end
end
```

```
%*********************************************************************
%    Addition of zeros to match origins (initial estimates)
%*********************************************************************

if origin_rows1a-origin_rows2a == 0
    IM2a = IM2a;
    Extra_Rows = 0;
elseif origin_rows1a-origin_rows2a > 0
    Extra_Rows = origin_rows1a-origin_rows2a;
    top = zeros(Extra_Rows,cols1);
    IM2a = [top; IM2a];
else
    Extra_Rows = origin_rows2a-origin_rows1a;
    bottom = zeros(Extra_Rows,cols1a);
    IM1a = [IM1a; bottom];
    IM2a = IM2a;
end
```

```
        [rows1 cols1] = size (IM2a);
        [rows1a cols1a] = size (IM1a);

            if origin_cols1a-origin_cols2a == 0
                Extra_Cols = 0;
            elseif origin_cols1a-origin_cols2a > 0
                Extra_Cols = origin_cols1a-origin_cols2a;
                left = zeros(rows1,Extra_Cols);
                IM2a = [left, IM2a];
            else
                Extra_Cols = origin_cols2a-origin_cols1a;
                left = zeros(rows1a,Extra_Cols);
                IM1a = [left, IM1a];
            end

%**********************************************************************
%      Addition of zeros to match matrix sizes (initial estimates)
%**********************************************************************

            [rows1a cols1a] = size(IM1a);
            [rows1b cols1b] = size(IM2a);

            if rows1a-rows1b > 0
                Extra_Rows = rows1a-rows1b;
                bottom = zeros(Extra_Rows,cols1b);
                IM2a = [IM2a; bottom];
            else
                Extra_Rows = rows1b-rows1a;
                bottom = zeros(Extra_Rows,cols1a);
                IM1a = [IM1a; bottom];
            end

        [rows1a cols1a] = size(IM1a);
        [rows1b cols1b] = size(IM2a);

            if cols1a-cols1b > 0
                Extra_Cols = cols1a-cols1b;
                right = zeros(rows1b,Extra_Cols);
                IM2a = [IM2a, right];
            else
                Extra_Cols = cols1b-cols1a;
                right = zeros(rows1a,Extra_Cols);
                IM1a = [IM1a, right];
            end
% figure(7),
% subplot(2,1,1)
% imagesc(IM1a)
% subplot(2,1,2)
% imagesc(IM2a)

% Image_mod3 = IM1a;
% Actual_image3 = IM2a;
%
% r = corr2(Image_mod3, Actual_image3)
```

```
%*********************************************************************
%        Determines the upper left corner of the two images
%*********************************************************************

if  horizontal_angle >= 0

            [rows2 cols2] = size (IM2a);

            for c = 1:cols2
                if sum(IM2a(:,c)) > 0
                    left2a_col = (c);
                    break
                end
            end

            for r = 1:rows2
                if (IM2a(r,left2a)) > 0
                    left2a_row = (r);
                    break
                end
            end

            [rows1 cols1] = size (IM1a);

            for c = 1:cols1
                if sum(IM1a(:,c)) > 0
                    left1a_col = (c);
                    break
                end
            end

            for r = 1:rows1
                if (IM1a(r,left1a)) > 0
                    left1a_row = (r);
                    break
                end
            end

%*********************************************************************
%        removes overlap which would reduce the correlation
%*********************************************************************

            if left1a_row - left2a_row > 0

                for h = 1:left1a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
                for h = 1:left1a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
```

```
                end
            end

        else

            for h = 1:left2a_row
                for Y = 1:cols1
                    IM1a(h,Y) = 0;
                end
            end
            for h = 1:left2a_row
                for Y = 1:cols2
                    IM2a(h,Y) = 0;
                end
            end
        end

% figure(8),
% subplot(2,1,1)
% imagesc(IM1a)
% subplot(2,1,2)
% imagesc(IM2a)

Else

%**************************************************************************
%        Determines the upper right corner of the two images
%**************************************************************************

        [rows2 cols2] = size (IM2a);

        for c = round(cols2/2):cols2
            if sum(IM2a(:,c)) == 0
                right2a_col = (c-1);
                break
            end
        end

        for r = 1:rows2
            if (IM2a(r,right2a_col)) > 0
                right2a_row = (r);
                break
            end
        end

        [rows1 cols1] = size (IM1a);

        for c = round(cols1/2):cols1
            if sum(IM1a(:,c)) == 0
                right1a_col = (c-1);
                break
            end
        end
```

```
            for r = 1:rows1
                if (IM1a(r,right1a_col)) > 0
                    right1a_row = (r);
                    break
                end
            end

%*********************************************************************
%        removes overlap which would reduce the correlation
%*********************************************************************

            if right1a_row - right2a_row > 0

                for h = 1:right1a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
                for h = 1:right1a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end

            else

                for h = 1:right2a_row
                    for Y = 1:cols1
                        IM1a(h,Y) = 0;
                    end
                end
                for h = 1:right2a_row
                    for Y = 1:cols2
                        IM2a(h,Y) = 0;
                    end
                end
            end

% figure(9),
% subplot(2,1,1)
% imagesc(IM1a)
% subplot(2,1,2)
% imagesc(IM2a)

end

% figure(6),
% subplot(4,2,8)
% imagesc(IM1a)
% subplot(4,2,7)
% imagesc(IM2a)
```

```
Image_mod3 = IM1a;
Actual_image3 = IM2a;

%*********************************************************************
%     Determines the correlation coefficient, and tracks the highest
%*********************************************************************

r = corr2(Image_mod3, Actual_image3);

        if r > rtrack(1,1)
            u = ((PixelVelx + x) / pixelsPerMeter) / dt;
            HR = (horizontal_angle);
            rtrack = [r; u; vel1;
                      HR; head_rate; head_rate1;
                      avgT; pitch; roll];
        end

% RTrack_plot = [RTrack_plot, rtrack];

end
end

% RTrack_plot1 = RTrack_plot';

        rtrack;
        % [correlation coefficient;
        %  U eststimate from the correlation coefficient;
        %  U calculated from the average of the instantaneous fwd
velocity;
        %  HR eststimate from the correlation coefficient;
        %  HR calculated from the change in compass heading over time;
        %  HR calculated from the average of the instantaneous stbd
velocity
        %  average time from when the sonar images were taken]

        % Velocity Estimates


%         if rtrack(2,:)>0.7 && rtrack(2,:)<2.0
%             vel = rtrack(2,:);
%         else
%             vel = 1.5;
%             disp('velocity est. out of bounds')
%         end

        vel = 1.0;
        Delta_head = 0; % rtrack(4,:);

t = toc;
    if t > MAXtime
        MAXtime = t;
    end
```

```
        RTrack = [RTrack, rtrack];

        Screen = [RTrack(1,:);  % correlation coefficient
                  RTrack(2,:);  % U est from correlation coefficient
                  RTrack(3,:);  % REMUS(l).estimatedVelocity averaged
                  RTrack(4,:);  % HR est from correlation coefficient
                  RTrack(6,:)]  % REMUS(m).headingRate averaged
        end
    end
end

for n = j+1:k+1

    if (REMUS(n-1).forwardVelocity) > -999
        fig(1,i) = REMUS(n-1).forwardVelocity;
        fig(2,i) = REMUS(n-1).altitude;
        fig(3,i) = REMUS(n-1).time;
        Fig = [Fig, fig];
    end
end

RTRACK = RTrack';
FIG = Fig';

%%
%************************************************************************
%       Total Plots
%************************************************************************

        figure,
        plot (RTrack(7,:), RTrack(1,:)), title('Correlation
coefficient')

        figure,
        hold on
        plot (RTrack(7,:), RTrack(2,:), '-r'),
        plot (RTrack(7,:), RTrack(3,:), '-k'),
        plot (RTrack(7,:), (RTrack(3,:)-RTrack(2,:)), '-b')
        plot (Fig(3,:)', Fig(1,:)', '-g')
        legend('Est U (Forward Velocity)', 'Calculated',...
               'Error', 'Measured Fwd Vel')
        title('Estimated and Measured U from REMUS 012506')

        figure,
        hold on
        % plot ( RTrack(7,:), RTrack(5,:), '-g'),
        plot (RTrack(7,:), RTrack(4,:), '-r'),
        plot (RTrack(7,:), RTrack(6,:), '-k'),
        legend('Est HR (Heading Rate)', 'Measured')
        title('Estimated and Measured HR from REMUS 012506')

        figure,
        plot (RTrack(7,:), (RTrack(4,:)-RTrack(6,:)), '-b')
```

```
        title('Error between Estimated and Measured HR from REMUS
012506')

%**********************************************************************
%       Analysis and Plots
%**********************************************************************
% After using Excel lookup to match times for Imagery-based velocity
% estimates to ADCP measured velocities and to calculate moving
% averages

load ('RTrack')

RTrack = RTrack';

Total_Error_U = mean(RTrack(3,:)-RTrack(2,:))
% Total_Error_V = mean(RTrack(4,:)-RTrack(6,:))

ABS_Error_U = mean(abs(RTrack(3,:)-RTrack(2,:)))
% ABS_Error_HR = mean(abs(RTrack(4,:)-RTrack(6,:)))

%**********************************************************************
%       Sub Plots
%**********************************************************************

a = 200;

for b = a:a:length(RTrack);
    c = b-(a-1);

        figure,
        subplot(2,1,1)
        hold on
        plot (RTrack(7,[c:b]), RTrack(2,[c:b]), '-','Color',[1 0.6
0.78]),
        plot (RTrack(7,[c:b]), RTrack(3,[c:b]), '-k', 'LineWidth',2),
        plot (RTrack(7,[c:b]), RTrack(10,[c:b]), '-',...
            'Color',[0.8471 0.1608 0], 'LineWidth',2)
        legend('Est U (Forward Velocity)', 'ADCP Measured U',...
            'Est U Moving Average', 'location', 'SouthOutside')
        title('Estimated and Measured U from REMUS 012506')
        xlabel('Time');
        xlim([RTrack(7,c), RTrack(7,b)]);
        ylabel('Velocity (m/s)');
        ylim([0.8, 2.2]);

        subplot(2,1,2)
        hold on
        plot (RTrack(7,[c:b]), (RTrack(3,[c:b])-RTrack(2,[c:b])), '-b')
        title('Error between Estimated and Measured U from REMUS
012506')
        xlabel('Time');
        xlim([RTrack(7,c), RTrack(7,b)]);
        ylabel('Velocity (m/s)');
```

```
        ylim([-1.0, 1.0]);

Average_Error_U = mean(RTrack(3,[c:b])-RTrack(2,[c:b]))
Average_ABS_Error_U = mean(abs(RTrack(3,[c:b])-RTrack(2,[c:b])))
Average_Error_HR = mean(RTrack(4,[c:b])-RTrack(6,[c:b]))
Average_ABS_Error_HR = mean(abs(RTrack(4,[c:b])-RTrack(6,[c:b])))

end
```

APPENDIX C: MATLAB CODE FOR OPENING FLS IMAGES

The following MATLAB® code was used for opening the individual sonar images within the simulations. The original code was developed by Doug Horner, with modifications made by M. Dolbec during the course of this work.

```
function sonarImage = OpenSonarImage(fileNumber)
% open a sonar image where the single argument is the file number
% it returns a two dimensional array of doubles



fileNumberStr = num2str(fileNumber);
%create the string that is the filename. File naming example is
%img-h1-p000002.raw

numLength = length(fileNumberStr);
if (numLength == 1)
    fileName1 = 'img-h1-p00000';
end
if (numLength == 2)
    fileName1 = 'img-h1-p0000';
end
if (numLength == 3)
    fileName1 = 'img-h1-p000';
end
if (numLength == 4)
    fileName1 = 'img-h1-p00';
end
if (numLength == 5)
    fileName1 = 'img-h1-p0';
end

fileNameExt = '.raw';

filename = strcat(fileName1,fileNumberStr,fileNameExt);
fullPathName = strcat('C:\DolbecImagesFirst\Allpings\',filename);

fid = fopen(fullPathName, 'r', 'b');

if fid == -1
    error('Failed to open file');
end

XSize = 334;
YSize = 464;

sonarImage = double(rot90(flipdim(fread(fid,[XSize YSize], ...
```

```
    '*uint16'),2)));
fclose(fid);
```

APPENDIX D: MATLAB CODE FOR SEGMENTATION LINES

The following MATLAB® code was used for creating lines connecting two pixels. The original code was developed by Georges Cubas, with modifications made by M. Dolbec during the course of this work.

```
function result = linept(matrix, X1, Y1, X2, Y2)
% Connect two pixels in a matrix with 1
%
% Command line
% ------------
% result=linept(matrix, X1, Y1, X2, Y2)
%   matrix : matrix where I'll write
%   (X1, Y1), (X2, Y2) : points to connect
%   result : matrix + the line
%
% Note
% ----
%   matrix can contents anything
%   (X1, Y1), (X2, Y2) can be out of the matrix
%
% Example
% -------
% a = linept(zeros(5, 10), 2, 2, 3, 9)
% a =
%
%      0     0     0     0     0     0     0     0     0     0
%      0     1     1     1     1     0     0     0     0     0
%      0     0     0     0     0     1     1     1     1     0
%      0     0     0     0     0     0     0     0     0     0
%      0     0     0     0     0     0     0     0     0     0
%
% Georges Cubas 20/11/03
% georges.c@netcourrier.com
% Version 1.0

result = matrix;
for x=max(1, X1):sign(X2 - X1):max(1, X2)
    y = round(f(x, X1, Y1, X2, Y2));
    if y > 0
        result(x, y) = -1;
    end
end
for y=max(1, Y1):sign(Y2 - Y1):max(1, Y2)
    x = round(f2(y, X1, Y1, X2, Y2));
    if x > 0
        result(x, y) = -1;
    end
end
```

```
function y=f(x, X1, Y1, X2, Y2)
a = (Y2 - Y1)/(X2 - X1);
b = Y1 - X1 * a;
y = a * x + b;

function x=f2(y, X1, Y1, X2, Y2)
if X1==X2
    x = X1;
else
    a = (Y2 - Y1)/(X2 - X1);
    b = Y1 - X1 * a;
    x = (y - b)/a;
end
```

LIST OF REFERENCES

Caccis, M., "Optical Triangulation-Correlation Sensor for Underwater Vehicles' Motion Estimation," *Proceedings of the 10th Mediterranean Conference on Control and Autonomation – MED2002*, Lisbon, Portugal, 2002.

Clark, Vernon, "Seapower 21, Projecting Decisive Force Capabilities", *United States Naval Institute Proceedings*, October 2002, www.usni.org, February 2007.

Cuschieri, J.; Negahdaripour, S., "Use of forward scan sonar images for positioning and navigation by an AUV," *OCEANS '98 Conference Proceedings* , vol.2, no.pp.752-756 vol.2, 28 September-1 October 1998.

Department of the Navy (2004). *The Navy Unmanned Undersea Vehicle (UUV) Master Plan.* Retrieved February 2007 from www.chinfo.**navy**.mil/navpalib/technology/**uuv**mp.pdf.

Fodrea, Lynn, "Obstacle Avoidance Control for the REMUS Autonomous Underwater Vehicle", Naval Postgraduate School, December 2002.

Fleischer, S.D.; Rock, S.M., "Experimental Validation of a Real-Time Vision Sensor and Navigation System for Intelligent Underwater Vehicles," *1998 International Conference on Intelligent Vehicles*, Stuttgart, Germany, October 1998.

Healey, Anthony J., Marco, D. B., "Slow Speed Flight Control of Autonomous Underwater Vehicles: Experimental Results with NPS AUV II," *Proceedings of the 2nd International Offshore and Polar Engineering Conference, San Francisco*, 14-19 July 1992.

Healey, Anthony J., *Dynamics of Marine Vehicles (MA-4823),* Class Notes, Naval Postgraduate School, Monterey, CA, 1995.

Healey, Anthony J., "Command and Control Demonstrations with Cooperating Vehicles", ONR Research Proposal in response to ONR BAA 01-012 "Demonstration of Undersea Autonomous Operation Capabilities and Related Technology Development", August 2001.

Horner, D.P.; Healey, A.J.; Kragelund, S.P., OCEANS, *AUV experiments in obstacle avoidance***,** Proceedings of MTS/IEEE, Vol., Iss., 2005Pages: 1464- 1470 Vol. 2, 2005.

Jain, R., Kasturi, R., and Schunck, B. *Machine Vision*. New York: McGraw-Hill, Inc., 1995.

Johnson, Jay, "Parameter Identification of the ARIES AUV," M.S. Thesis Naval Postgraduate School, Monterey, CA, June 2001.

Jorgensen, K.V.; Grose, B.L.; Crandall, F.A., "Doppler sonar applied to precision underwater navigation," *OCEANS '93. 'Engineering in Harmony with Ocean'. Proceedings* , vol., no.pp.II469-II474 vol.2, 18-21 October 1993.

Kalyan, B.; Balasuriya, A., "Multiple sensors based navigation scheme for AUV position estimation," *Underwater Technology, 2004. UT '04. 2004 International Symposium on* , vol., no.pp. 201- 207, 20-23 April 2004.

Leonard, J.J., and Newman, P., "Consistent, Convergent, and Constant-Time SLAM", in *Proceedings 18th International Joint Conference on Artificial Intelligence 2003*, Morgan Kaufmann, pp. 1143-1150, 2003.

Marine Navigation. 23 March 2006. NOAA's National Ocean Service. February 2007. http://www.oceanservice.noaa.gov/topics/navops/marinenav/welcome.html.

Marco, D.B.; A.J. Healey, "Command, Control and Navigation Experimental Results with the NPS ARIES AUV," *IEEE Journal of Oceanic Engineering* – Special Issue, 2001.

Prestero, Timothy, "Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle," M.S. Thesis Massachusetts Institute of Technology, September 2001.

Roumeliotis, S.I.; Johnson, A.E.; Montgomery, J.F., "Augmenting inertial navigation with image-based motion estimation," *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on* , vol.4, no.pp. 4326-4333 vol.4, 2002.

Smith, R.; Self, M.; Cheeseman, P., "Estimating Uncertain Spatial Relationships in Robotics", *Autonomous Robot Vehicles*, I. J. Cox and G. T. Wilfong, Eds. Springer-Verlag New York, New York, NY, 167-193. 1990.

Sonka, M., Hlavac, V., Boyle, R., *Image Processing, Analysis and Machine Vision*. London, England: Chapman and Hall, 1995.

Thompson, L.R.; Seawall, J.; Josserand, T., "Two Dimensional and Three Dimensional Imaging Results Using Blazed Arrays", *OCEANS, 2001 MTS/IEEE Conference and Exhibition*, Vol.2, Iss., Pages:985-988 vol.2. 2001.

Waite, A.D., *Sonar for Practising Engineers, Third Edition*. West Sussex, England: John Wiley and Sons, Ltd, 2002.

BIBLIOGRAPHY

Blidberg, Richard D., "The Development of Autonomous Underwater Vehicles (AUVs); A Brief Summary", Autonomous Undersea Systems Institute, ICRA, Seoul, Korea, May 2001.

Dudek, Gregory, and Jenkin, Michael, *Computational Principles of Mobile Robotics*, Cambridge University Press, United Kingdom, 2000.

Forsyth, David A., and Ponce, Jean, *Computer Vision: A Modern Approach*, Prentice-Hall, Inc, Upper Saddle River, New Jersey, 2002.

Healey, A. J., An, E. P., and Marco, D.B., "Online compensation of heading sensor bias for low cost AUVs," *Autonomous Underwater Vehicles, 1998. AUV'98. Proceedings Of The 1998 Workshop on* , pp.35-42, 20-21 August 1998.

Leonard, J.J., Carpenter, R. N., and Feder, H. J. S., "Stochastic Mapping Using Forward Look Sonar," in *Proceedings International Conference Field and Service Robotics*, Pittsburgh, PA, pp. 69–74, August 1999.

Murphy, Robin R., *Introduction to AI Robotics*, The Massachusetts Institute of Technology Press, 2000.

Ogata, Katsuhiko, *Modern Control Engineering, Fourth Edition*, Prentice-Hall, Inc, Upper Saddle River, New Jersey, 2002.

Se, S., Lowe, D.G., and Little, J.J., "Vision-based global localization and mapping for mobile robots",. *Robotics and Automation, IEEE Transactions on,* pp. 364- 375, Volume: 21, Issue: 3, June 2005.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Fort Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Mechanical Engineering Department Chair, Code ME
   Distinguished Professor Anthony J. Healey
   Naval Postgraduate School
   Monterey, California

4. Mechanical Engineering Curriculum, Code 34
   Naval Postgraduate School
   Monterey, California

5. Undersea Warfare Department Chair, Code USW
   Professor Donald Brutzman
   Naval Postgraduate School
   Monterey, California

6. Undersea Warfare Curriculum, Code 75
   Naval Postgraduate School
   Monterey, California

7. Submarine Development Squadron Twelve
   Naval Submarine Base New London
   Groton, Connecticut

8. Dr. Tom Swean, Code 32
   Office of Naval Research
   Arlington, Virginia

9. Pierre J. Corriveau, Chief Technology Officer
   Naval Undersea Warfare Center
   Newport, Rhode Island

10. Professor Douglas Horner
    Naval Postgraduate School
    Monterey, California

11. Professor Mathias Kolsch
Naval Postgraduate School
Monterey, California

12. LT Michael Dolbec
Naval Submarine Base New London
Groton, Connecticut